Universidad Carlos III de Madrid

ESCUELA POLITÉCNICA SUPERIOR

DEPARTAMENTO DE INGENIERÍA DE SISTEMAS Y AUTOMÁTICA

Proyecto Fin de Carrera

μMote: Interfaz hombre–máquina inalámbrica de control para robótica asistencial

Autor: Jorge Jiménez Rodríguez

Tutor: Dr. Juan Carlos González Víctores

Director: Dr. Alberto Jardón Huete

INGENIERÍA TÉCNICA INDUSTRIAL : ELECTRÓNICA INDUSTRIAL

LEGANÉS, MADRID

JULIO 2015

UNIVERSIDAD CARLOS III DE MADRID

DEPARTAMENTO DE INGENIERÍA DE SISTEMAS Y AUTOMÁTICA

PROYECTO FIN DE CARRERA

El tribunal aprueba el Proyecto Fin de Carrera titulado *"μMote: Interfaz hombre–máquina inalámbrica de control para robótica asistencial"*, realizado por Jorge Jiménez Rodríguez y dirigido por Dr. Juan Carlos González Víctores.

Firma del Tribunal Calificador:

PRESIDENTE: Santiago Martínez de la Casa ..... _____

SECRETARIO: José Medina Hernández ..... _____

VOCAL: Susana Patón Álvarez ..... _____

CALIFICACIÓN: _____

Leganés, Madrid, a 27 de julio de 2015.

A mi padre y a mi madre

A mi hermana

Agradecimientos

En primer lugar, quiero dar las gracias a Juan Carlos González Víctores, por la gran ayuda, apoyo y consejos que me ha brindado en la realización del proyecto, ha sido un excelente tutor. Esta experiencia me ha hecho aprender mucho y me ha servido para crecer tanto profesional como académicamente. Ha sido un placer ser guiado por alguien tan cercano, humano y con un conocimiento tan extenso sobre las materias a tratar. Agradecerle además la paciencia que ha tenido conmigo todo este tiempo.

Dar las gracias también a Alberto Jardón Huete por permitirme participar en el Departamento de Ingeniería de Sistemas y Automática, y dejar que aportara mi granito de arena en parte de los proyectos de investigación dirigidos por él. Como tutor en el trabajo dirigido previo a este proyecto fin de carrera, como iniciador del presente tutelaje y como profesor en la carrera, pude ver su visión y su enfoque respecto a los problemas tecnológicos y cómo los aborda. Ese aprendizaje siempre lo llevaré conmigo.

Agradecer es una palabra que se queda corta para expresar mi deuda eterna con mi familia. Yo lo soy todo gracias a ellos. Hemos pasado muchas dificultades de diversa índole y aún así siempre ha sido prioritario que los hijos pudiéramos estudiar y tener la mejor formación posible. La consecución final académica ha llevado más tiempo del deseado, pero eso no quita la realidad conseguida gracias a mi familia desde hace mucho, ser una persona buena, humana y con valores. Por todo ello, a mi padre, a mi madre, a mi hermana, gracias.

Todo mi agradecimiento también, a mis amigos y amigas, los cercanos, los de verdad, los que de una manera u otra me han ayudado y apoyado para terminar este proyecto. Todos saben que terminar esto es muy importante para mí, se cierra una etapa y se abre otra nueva. Hacer especial mención a mis amigos David y Jorge, por estar siempre ahí, en los momentos buenos, en los regulares y en los malos. Gracias.

Por último, agradecer al lector del presente documento que no esté citado más arriba. Espero que su lectura sea lo más agradable posible. Bienvenid@.

Resumen

El presente proyecto fin de carrera se enmarca dentro del ámbito de los dispositivos de control para robótica asistencial. Se busca desarrollar una interfaz que facilite a las personas discapacitadas y personas mayores el control del robot. Es importante la mejora continua de los robots asistenciales para facilitar las tareas cotidianas y proporcionar una mayor independencia a estas personas, pero también lo es la mejora en los dispositivos de control, intentando que éstos sean cada vez más sencillos y fáciles de utilizar.

En este sentido la interfaz pretende simplificar el control del movimiento del robot, utilizando como mecanismo el propio movimiento de la persona. El dispositivo irá acoplado de forma solidaria al usuario, en la extremidad o parte del cuerpo que tenga mayor movilidad. Así, se pretende replicar el movimiento que efectúe la persona, en el robot.

Para ello se ha realizado un diseño hardware en el que se integra en una misma placa de circuito impreso (PCB), un acelerómetro que detecta los cambios de inclinación en cada uno de los tres ejes espaciales, un microcontrolador que gestiona toda la información y un módulo inalámbrico que envía los datos obtenidos a un ordenador. El ordenador tiene que ser capaz de recibir esa información del movimiento de la PCB, mostrarla en pantalla, y en un futuro desarrollo, trasladarla al robot.

En cuanto a la programación, se debe escribir un código para el microcontrolador y otro código para ejecutar en el ordenador, que consigan la funcionalidad buscada.

Se diseña también un soporte físico para proteger y contener toda la electrónica de la PCB.

El dispositivo deberá ser pequeño, ligero, autónomo, barato y fácil de usar. También es importante la libertad de movimientos y por eso el diseño incluye de base la característica de ser inalámbrico. Todo esto con la intención de que el control del robot sea lo más intuitivo y sencillo posible.

Abstract

This final degree project falls within the scope of control devices for assistive robotics. It seeks to develop an interface that facilitates robot control for disabled and elder people. Continous development in assistive robots is important in order to achieve easier daily tasks and to provide greater independence to these people, but it is also important to improve control devices, trying that they are more simple and easy to use.

In this sense, the interface pretends to simplify robot motion control, using the own person movement as mechanism. The device will be connected in solidarity to the user, in the end or body part which has more mobility. So, to replicate the person movement in the robot is pretended.

A hardware design has been done for that, in what in a same printed circuit board (PCB) is integrated one accelerometer that detects tilting changes in each of the three spatial axes, one microcontroller that manages all the information, and one wireless module that sends the obtained data to the computer. The computer must be able to receive that PCB motion information, display it on screen, and in a future development, transfer it to the robot.

In regard to programming, one code should be written for the microcontroller and another code should be written to run on the computer, so that both can achieve the desired functionality.

A physical mounting for the PCB is also designed in order to protect and contain all the electronics.

The device will be small, lightweight, autonomous, cheap and easy to use. Freedom of movement is also important and that is why the design includes wireless feature from the beginning. All this with the intention of achieving the simplest and most intuitive robot control.

Índice general

Índice de Figuras

Índice de Tablas

Capítulo 1

Introducción

La tecnología es una invención del hombre para ayudar al hombre. Es notorio el impacto que los avances tecnológicos han tenido en el mundo, y el siglo XXI parece ser la puerta de entrada a una vorágine de transformaciones científicas y tecnológicas que llevarán las relaciones entre las máquinas y los seres humanos a otro nivel. Parece claro que el ámbito de la robótica tendrá un papel fundamental en ese futuro y está en nuestra mano aportar el conocimiento y el trabajo necesario para ayudar a construir lazos entre los más avanzados proyectos de investigación académicos y el aprovechamiento de la robótica para el uso cotidiano.

Es en ese sentido en el que se encuadra el presente proyecto fin de carrera. La intención de ayudar a las personas con el desarrollo de un dispositivo tecnológico que sirve de nexo entre un sistema robótico avanzado y la persona que lo pretende manejar. Se está por tanto, ante una interfaz hombre–máquina (HMI). Un intermediario para facilitar la comunicación entre la persona y la máquina.

μMote: Interfaz hombre–máquina inalámbrica de control para robótica asistencial. El título del proyecto indica que el tipo de robótica para el que se desarrolla la interfaz es el asistencial. A diferencia de la robótica orientada a la industria o la manufactura, la robótica asistencial aparece en el entorno doméstico para ayudar a las personas en las tareas cotidianas, y para ello se han desarrollado los llamados robots asistenciales [1]. Entre otros colectivos, estos robots prestan ayuda a las personas con discapacidad y a las personas mayores. La falta de movilidad y la imposibilidad de realizar determinadas tareas que tienen estas personas les confiere una dependencia de otros, que hace que la irrupción de robots asistenciales en sus vidas les pueda facilitar las cosas y darles, si no toda, sí una cierta independencia muy deseada por ellos.

Para controlar y dirigir estos robots, muchas veces la tarea no es sencilla, ya que o bien la discapacidad o la falta de movilidad concreta de la persona no permite manejar el dispositivo de control, o bien el mismo dispositivo de control es demasiado complejo de utilizar para estas personas. Es por ello que es importante no sólo ir mejorando con el tiempo los robots asistenciales, sino también ir mejorando el dispositivo o la interfaz de control del robot.

Las características más adecuadas para un dispositivo de control van a depender tanto del robot concreto que se maneja como del tipo de persona que tiene ese control, pero sí se pueden nombrar algunos aspectos comunes que generalmente se intentan buscar. Facilidad de uso, bajo peso, pequeño tamaño, bajo consumo, y por tanto, mayor autonomía. En este tipo de aplicaciones asistenciales es claro que lo más importante es que sea fácil de usar, que sea sencillo y sin limitaciones físicas. Un dispositivo de control puede estar muy bien hecho pero si depende de cables para estar alimentado o para enviar información, limita mucho físicamente la movilidad y la independencia. Por ello, una característica prioritaria de este proyecto ha sido que la interfaz sea inalámbrica, sin cables. Asimismo, la facilidad de uso se ha constituido en característica intrínseca al proyecto. Y el bajo peso, bajo consumo y dimensiones reducidas se han considerado objetivos a conseguir.

El nombre de μMote, o bien, microMote, se escoge para nombrar de una forma sencilla y rápida a la interfaz. Este nombre se inspira en el Wiimote$^{\text{TM}}$, que es el mando inalámbrico de control de la consola de videojuegos Wii de Nintendo$^{\text{TM}}$. Y teniendo en cuenta que, por un lado, para el desarrollo de la interfaz se utilizan conocimientos en microelectrónica, y por otro lado, se pretende diseñar un dispositivo mucho más pequeño de lo habitual, ha sido adecuado escoger el prefijo 'micro' (μ).

La interfaz en sí misma consiste en un dispositivo que detecta los movimientos de la persona que lo utiliza, y transfiere la información de esos movimientos de forma inalámbrica a un ordenador, el cual es capaz de recibir y mostrar esos datos en pantalla. Además, esta información es trasladada al robot para que replique los movimientos efectuados por la persona, aunque este último apartado se concretará en desarrollos futuros. Por tanto, esta interfaz está indicada para facilitar el uso de robots asistenciales que pretendan para su movimiento propio la réplica de los movimientos de la persona que lo controla. Ver figura 1.1.

La captación del movimiento por parte de la interfaz se va a llevar a cabo mediante un acelerómetro o sensor inercial, y el tratamiento y gestión de la información medida por el sensor lo va a realizar un microcontrolador. Éste a su vez, mandará los datos a través de un módulo inalámbrico (que será de tecnología Bluetooth), para que los reciba un ordenador o PC. El ordenador será capaz de recibir y entender esa información.

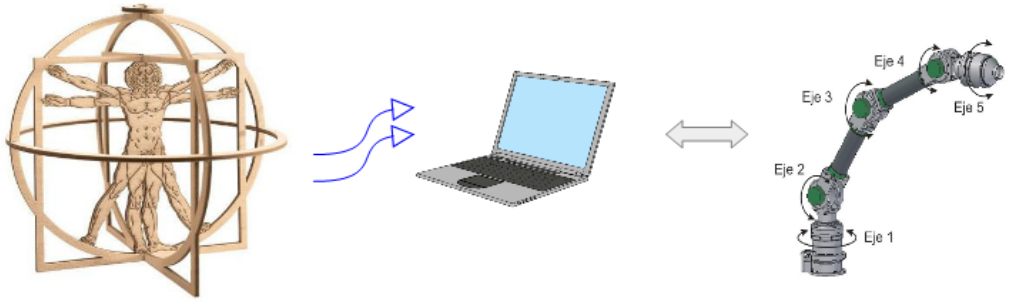


Figura 1.1: Propósito de la interfaz hombre–máquina (HMI)

Para ejecutar lo anterior se va a desarrollar un diseño hardware que busca integrar en una misma placa de circuito impreso (PCB) toda la electrónica utilizada. A su vez, se va a desarrollar un diseño software para programar el microcontrolador que hace de cerebro gestor de todo el dispositivo, y se va a escribir también un programa para el ordenador, que hará de receptor de la información generada en la PCB. Para proteger los componentes electrónicos de la PCB y facilitar el uso del dispositivo por parte de la persona que lo utilice, se va a diseñar un soporte físico que lo contenga.

Por tanto, la interfaz inalámbrica de control requiere de un diseño hardware, software y mecánico simultáneo que se desarrolla a lo largo de todo el documento.

1.1. Motivación

El proyecto surge de la necesidad por parte del Departamento de Sistemas y Automática de la Universidad de hacer una versión 2.0 del dispositivo apuntador realizado por Verónica López Vaquero en su proyecto fin de carrera "Dispositivo inalámbrico para facilitar el acceso al ordenador" [2]. El tutor de ese proyecto, Alberto Jardón Huete, es el profesor e investigador del departamento responsable y creador del robot ASIBOT [1]. Éste es un robot asistencial que consiste en un brazo robótico diseñado para facilitar las tareas cotidianas de las personas con discapacidad o personas mayores. Se pretende conseguir una mejora sustancial en el dispositivo de control del robot, ya que las HMI utilizadas hasta el momento o bien eran muy complejas de utilizar como la PDA, o muy aparatosas y grandes como los mandos con joystick cableados.

Entonces la idea es conjuntar todo lo anterior, y realizar una interfaz inalámbrica que controle el robot, pero lo más pequeña y ligera posible, y sobre todo

intuitiva y fácil de usar.

Aunque la aplicación directa de la interfaz es para el robot ASIBOT, esta interfaz puede utilizarse para cualquier otro robot asistencial ya que el desarrollo de este proyecto se queda en la recepción y muestra en pantalla por parte del ordenador de la información del movimiento de la persona.

Es por tanto, intención y motivación de este proyecto, el desarrollar un dispositivo tecnológico que sea útil en la práctica, y facilite en este caso el control del robot asistencial por parte de las personas discapacitadas o las personas mayores. Teniendo en cuenta que además, la evolución demográfica y los avances médicos indican que cada vez habrá más población de avanzada edad, ello implica que en unos años habrá muchas más personas mayores que demandarán mayor independencia y calidad de vida. En este punto, ayudar a desarrollar proyectos que faciliten la vida a esas personas es interesante y motivador.

El desarrollo de esta interfaz con estas características se propone también como una búsqueda de una alternativa más barata, más cómoda y menos invasiva, a otros sistemas de control robótico muy eficientes que se están desarrollando hoy en el mundo pero que requieren por ejemplo de intervención quirúrgica o aplicación de electrodos. La interfaz de este proyecto pretende que sea tan sencillo y fácil como ponerse un reloj de pulsera.

1.2. Objetivos

Los objetivos principales que se pretenden conseguir son:

- Tamaño lo más pequeño posible.
- Bajo consumo, y por tanto, autonomía eléctrica lo más alta posible.
- Bajo precio.
- Peso ligero.
- Facilidad de uso.

Es imprescindible que la independencia o autonomía del dispositivo sea total y no dependa de cables, por ello, es característica previa que el dispositivo sea inalámbrico.

Además, se pretende que la universalidad del producto sea lo mayor posible, es decir, que la interfaz pueda ser usada en la mayor cantidad de lugares, respecto a la tecnología a utilizar.

Una característica previa para el hardware es que toda la electrónica se integre en una sola placa de circuito impreso (PCB). Por supuesto, el software que se programe en el microcontrolador tiene que conseguir realizar correctamente los procesos para llevar la información medida del acelerómetro al ordenador. Y el software que se programe en el PC tiene que ser capaz de recibir y mostrar esa información.

1.3. Estructura del documento

El documento describe de manera secuencial el desarrollo del proyecto.

El capítulo 2 hace una descripción sobre el estado tecnológico existente en las áreas principales sobre las que se basa la interfaz, que son los dispositivos inalámbricos de control, las interfaces hombre–máquina en robótica asistencial, los sensores de aceleración–inclinación y la tecnología Bluetooth.

El capítulo 3 presenta la interfaz del proyecto explicando su funcionalidad y propósito, así como las especificaciones de diseño que se tienen en cuenta para su desarrollo. También se detalla la selección de los componentes electrónicos que se van a utilizar.

El capítulo 4 introduce el diseño hardware del dispositivo, empezando con el diseño teórico del circuito, y continuando con su construcción física primero en una placa experimental para realizar pruebas y ensayos, y después con el diseño y construcción de la placa de circuito impreso (PCB) definitiva.

El capítulo 5 trata el proceso de desarrollo del código para programar el microcontrolador y el ordenador, realizando una descripción secuenciada que explica el software definitivo al que se termina llegando.

El capítulo 6 describe el diseño en 3D del soporte físico del dispositivo, así como el material y maquinaria necesarios para imprimirlo.

El capítulo 7 explica las pruebas y ensayos realizados en la placa experimental, que dan paso a la puesta en funcionamiento de la interfaz definitiva con ensayos para comprobar la correcta funcionalidad. También se describe la integración de la PCB con el soporte físico.

El capítulo 8 presenta las conclusiones y la reflexión sobre el cumplimiento de los objetivos. Además, se describen ampliaciones y desarrollos futuros que por un lado concluyen el trabajo iniciado aquí, y por otro presentan posibilidades de mejora futuras.

En los apéndices se muestran apartados muy importantes para el seguimiento de la información descrita en los capítulos.

El presente documento está redactado en LaTeX[3].

Capítulo 2

Estado del Arte

En este capítulo se va a hacer un repaso al estado actual de la tecnología relacionada con los aspectos principales del proyecto. Primero, haciendo una revisión de los dispositivos de control existentes que utilizan tecnología inalámbrica, a nivel general, para cualquier tipo de uso. Después se va a describir, dentro ya de la robótica asistencial, qué tipo de interfaces hombre–máquina (HMI) se pueden encontrar hoy en día, sin tener en cuenta su dependencia o no de cables. Como el componente básico de la interfaz para medir los movimientos es un acelerómetro, se van a repasar los tipos de sensores de aceleración–inclinación que hay para encuadrar la opción elegida. Y por último, se hace un breve estudio comparativo entre varias tecnologías inalámbricas para justificar la elección del tipo de módulo inalámbrico, y se muestran las características principales de la tecnología Bluetooth junto con algunos dispositivos actuales.

2.1. Dispositivos inalámbricos de control

Cualquier aparato tecnológico que requiera órdenes de acción por parte de la persona que lo maneje va a necesitar de un dispositivo de control para llevar esas órdenes al sistema. En los últimos años se está popularizando que ese control sea llevado por dispositivos inalámbricos, sin cables, ya que permiten una mayor movilidad y comodidad pues no se tiene la limitación física de tener que estar o bien a la distancia máxima del cable o bien a distancia cero manipulando el mismo aparato en sí. Se estima que el número de personas en el mundo que usan dispositivos inalámbricos ha pasado de unos 100 millones en el año 2000 a aproximadamente 500 millones en 2010 [4].

Existe una gran variedad de dispositivos inalámbricos orientados al control.

Se va a referir la manejabilidad de cada uno.

- Ratón y teclado inalámbricos. Periféricos asociados al ordenador que sirven como dispositivos de entrada a la máquina. Requieren buen manejo de manos y dedos.
- Tablet. Dispositivo táctil más intuitivo en el control que también requiere buen manejo de manos y dedos.
- PDA. Del inglés, asistente digital personal. Lo más característico de una PDA es su portabilidad y su capacidad de estar abierta al desarrollo software para otorgarle una función específica y solucionar un problema concreto [5]. La combinación de pantalla con lápiz táctil y botones requiere igualmente de un buen manejo de manos y dedos.
- Smartphone. Junto con la PDA y la tablet, el smartphone es un tipo de dispositivo que se puede programar para controlar otros sistemas. Con una aplicación dedicada a la tarea concreta, desde un smartphone se pueden controlar desde aparatos domésticos hasta sistemas más complejos. Hay aplicaciones disponibles comercialmente y también proyectos de desarrollo para control remoto de dispositivos con Bluetooth 4.0 [6]. Al ser táctil, su manejo está orientado a las manos.
- Mando de videojuegos inalámbrico. Las consolas de videojuegos necesitan de un dispositivo de control para recibir las órdenes de acción del usuario. La primera patente de un mando inalámbrico para videojuegos data del año 1985 [7]. La interacción se basa en botones que envían una señal de radiofrecuencia a la consola central. No es hasta la aparición de la consola Wii de Nintendo™ con su mando Wiimote™ en 2006 que cambia el concepto de mando de control en este ámbito, añadiendo a los botones la capacidad de detectar el movimiento del usuario gracias a un acelerómetro incorporado. Por supuesto el manejo es con las manos.
- Mandos de radiofrecuencia (RF). Aquí los dispositivos más extendidos son los controles remoto por infrarrojos (IR), prácticamente todos los mandos de aparatos domésticos son de este tipo [8]. También son muy utilizados para aplicaciones ad hoc, por ejemplo en robótica se puede utilizar el mando de la empresa Sparkfun (figura 2.1) para controlar un robot simple, o bien, para tener un control más complejo utilizar el mando IR de la compañía Makeblock; figura 2.2 [9]. Al constar de botones para el manejo, de nuevo, se utilizan dedos y manos.
- Control remoto para la industria. Aunque basados la mayoría en RF, la industria ha desarrollado multitud de soluciones de control inalámbrico que se han implementado en fábricas y empresas de todo el mundo. Soluciones robustas y contrastadas, lo cual merece un comentario aparte.

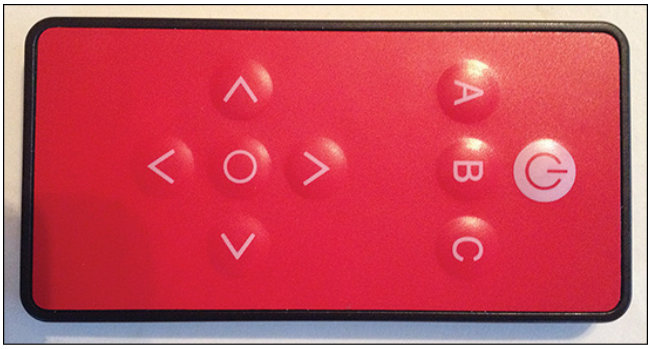


Figura 2.1: Mando de control RF de Sparkfun

Figura 2.2: Mando de control RF de Makeblock

El manejo manual se impone de nuevo. Además, la tecnología inalámbrica en sistemas industriales ofrece muchas posibilidades de mejora en el control de procesos. Como ejemplo, la implementación de un sistema de comunicación inalámbrica a través de un PC, PLC y SCADA, mediante tecnología Bluetooth [10].

- Otros. Entre otros muchos dispositivos de control inalámbricos, se pueden citar dispositivos de control óptico por emisión de diodos LED [11], pizarras digitales interactivas [12] o redes de sensores para control de generadores de vapor [13].

Como se ha podido ver, casi todos los dispositivos de control requieren de las manos y de los dedos con una funcionalidad normal para poder utilizarlos. Esto limita el uso a personas con problemas motores en estas extremidades, y por ello se plantean soluciones como la interfaz de este proyecto.

2.2. Interfaces hombre–máquina en robótica asistencial

En la robótica asistencial es muy importante el nexo entre la persona y la máquina, puesto que estamos ante personas con limitaciones, y por tanto van a tener más dificultad para manejar o controlar un sistema robótico. Precisamente es este último el que pretende facilitar la vida a estas personas, y es ahí donde el diseño tanto de los robots como de las interfaces de control tiene que hacerse con más atención [14].

Las interfaces hombre–máquina (HMI) en el ámbito asistencial dependen mucho del tipo de asistencia que se quiere dar, y aunque algunas son interfaces genéricas, muchas otras son indicadas para un tipo muy concreto de circunstancia de la persona.

Veamos algunas HMI para robótica asistencial.

- Ratón Bluetooth para personas con discapacidad. Joystick adaptado a una silla de ruedas que a modo de ratón de ordenador basa su funcionamiento en un acelerómetro, un microcontrolador y un módulo Bluetooth. Claro antecedente funcional del presente proyecto, pero sus componentes analógicos y montaje aparatoso hacen que sea un sistema rudimentario y poco manejable [15].

- Joystick inalámbrico para facilitar el acceso al ordenador. Dispositivo basado en un acelerómetro, un microcontrolador y conexión Wi-Fi, acoplado a una silla de ruedas para facilitar a la persona discapacitada el manejo del ordenador. Mediante el movimiento del joystick adaptado se puede simular el movimiento de un ratón de ordenador. El acelerómetro, que es analógico, está situado en el joystick, y a través de un cable envía la información medida a una caja donde se encuentra el resto de la electrónica. Figura 2.3. Este dispositivo es la referencia principal de la que parte este proyecto [2] [16].

Figura 2.3: Dispositivo inalámbrico para facilitar el acceso al ordenador

- Tablet. Se pueden desarrollar aplicaciones para tablet-PC orientadas al manejo de robots. Se ha implementado una interfaz realizada en Flash para el robot Maggie de la Universidad Carlos III de Madrid, la cual facilita la comunicación entre la persona y la máquina a través de una serie de menús y secciones sencillos e intuitivos. Figura 2.4 [17].

- PDA. La adaptabilidad en la programación y la portabilidad de las PDAs las hace muy indicadas para hacer de intermediario entre el robot y el usuario. En el ámbito sanitario las PDAs están muy extendidas tanto en la formación de profesionales como en la atención a pacientes [18], y se hace extensible su utilización para la asistencia de estos pacientes en entornos tecnológicos. El robot asistencial ASIBOT de la Universidad Carlos III de Madrid puede controlarse con una PDA, que es una de las partes del sistema asistencial de la figura 2.5 [1].

- Interfaz de joystick para robot. El robot ASIBOT tiene como una de las interfaces para controlar el robot el joystick que se puede ver en la figura 2.6. Dispone de dos modos de funcionamiento, movimientos preprogramados o movimientos teleoperados, de tal manera que se puede asignar una tarea concreta al robot ya programada, o bien moverlo de forma libre [19].

Figura 2.4: Aplicación Flash para tablet en el robot Maggie

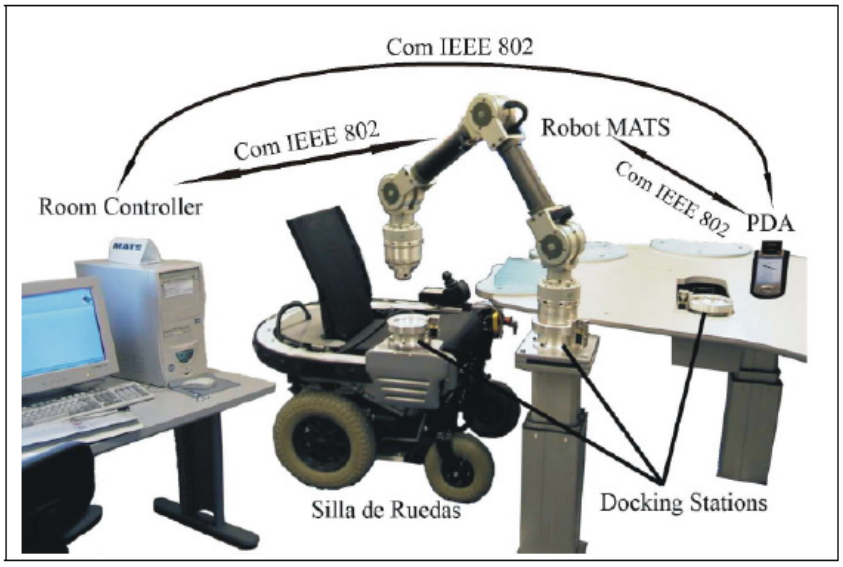


Figura 2.5: Sistema asistencial del robot ASIBOT, incluyendo la PDA

Figura 2.6: Joystick de control para el robot ASIBOT

- Wiimote™. El mando de la videoconsola Wii de Nintendo™ tiene un acelerómetro en su electrónica que le permite detectar los movimientos del usuario en los ejes x, y, z; figura 2.7. En 2008 aparece una mejora llamada Wii Motion Plus que consiste en un sensor giroscópico que detecta los giros de rotación y elevación (roll y pitch). Uno de los inconvenientes del Wiimote™ es que necesita una barra de LEDs infrarrojos receptora de datos instalada enfrente del usuario, por lo que el mando no es autónomo para ubicar su situación exacta. Hay un desarrollo en la Universidad Carlos III que utiliza el Wiimote™ como control del robot ASIBOT [20].

Figura 2.7: Mando de control Wiimote™

- Sensores. Otros HMI para robótica asistencial son directamente sensores inerciales, como el sensor MTi, que es un sensor con magnetómetros 3D integrados (figura 2.8). Con referencia al sistema de coordenadas magnéticas de la Tierra, el sensor es capaz de detectar su posición exacta en cada

momento. Esta información se puede trasladar al robot para inferirle un movimiento determinado [21].

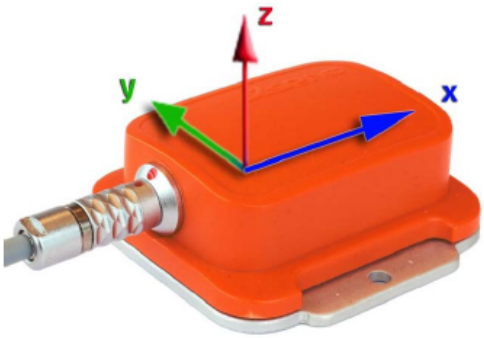


Figura 2.8: Sensor inercial MTi

- Indicadores de cabeza. Hay interfaces desarrolladas expresamente para personas parapléjicas que sólo tienen movilidad en la cabeza. Existen diversas tecnologías. Unos toman como base dispositivos optoelectrónicos y comunicación infrarroja como la interfaz de la figura 2.9 [22]. Otros basan su funcionamiento en acelerómetros y giroscopios [23]. Todos ellos adaptando el soporte para ser utilizado en la cabeza ya sea por cinta de sujeción o estructura facial.

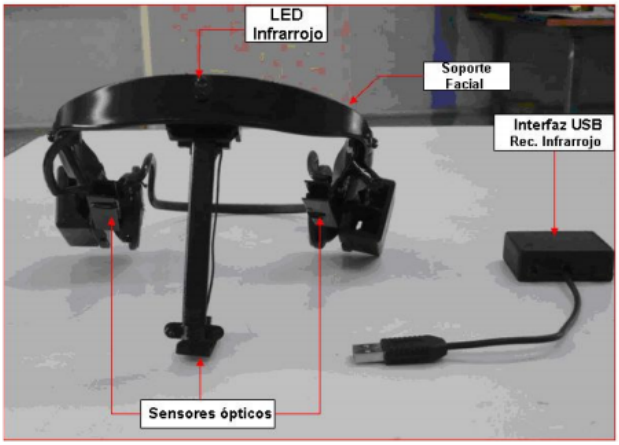


Figura 2.9: HMI indicador de cabeza

- Voz. Para que la persona pueda comunicar con un robot hay un método muy sencillo para la persona que es hablar. Aquí la dificultad está en que el robot entienda lo que le queremos decir. Hay que tener en cuenta que, obviamente, para personas con una discapacidad o limitación en el habla, este tipo de solución no se puede usar. Se están llevando a cabo muchos proyectos de investigación en el mundo que usan la voz como HMI. En

la Universidad Carlos III el robot asistencial Maggie se puede comunicar con el habla. También hay proyectos para enseñar a los robots, hacer que aprendan vocabulario y sean capaces de reconocer objetos que no habían visto anteriormente. Y entre otros métodos se puede usar la voz para aportar información al robot. Esta investigación se ha realizado con el robot Teo, el cual se puede ver en la figura 2.10 [24].

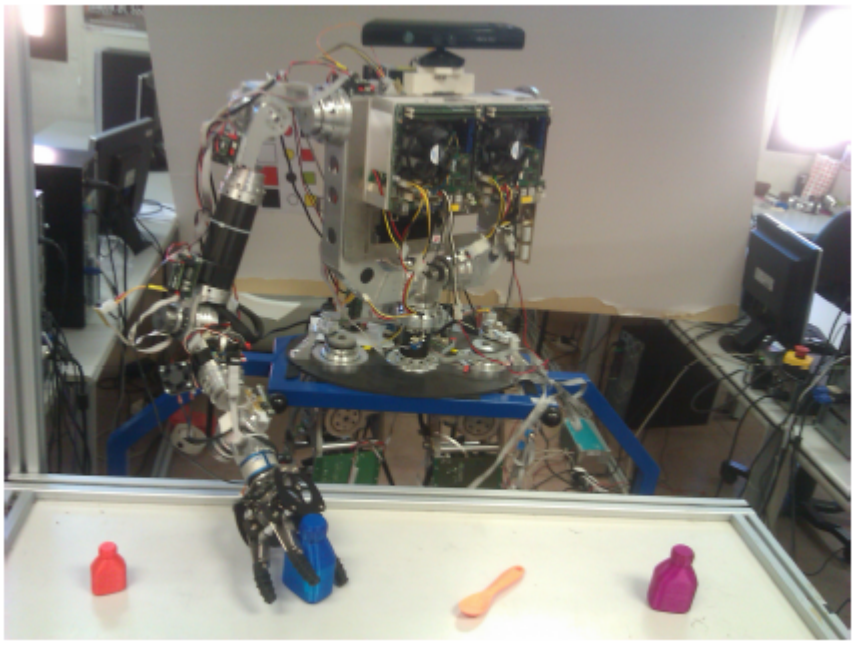


Figura 2.10: Robot Teo. HMI por voz

- Ingeniería software. En robótica asistencial es importante la distinción entre HMIs orientadas al desarrollador u orientadas al usuario, ya que estas últimas tienen que ser especialmente sencillas y con el objeto de controlar sólo funcionalidades de alto nivel [25]. En este sentido todo el desarrollo software tiene que ser flexible para tener buena intercompatibilidad entre distintos entornos y arquitecturas. Muchas interfaces para el usuario son entornos gráficos (GUI) programados para dispositivos móviles (PDA, tablet, smartphone, etc.) y por ello se puede considerar el software como una HMI.

- Control cerebral. Las interfaces cerebro–computador (BCI) se basan en la adquisición de señales de electroencefalograma (EEG) mediante una red de electrodos distribuidos en puntos clave alrededor del cerebro. Hay sistemas invasivos (con operación quirúrgica) y no invasivos (casco o red de sensores). Aplicado a la robótica asistencial hace que el control de la persona sobre el robot sea totalmente directo a través de la actividad cerebral. El inconveniente principal es la incomodidad del sistema. Además, las investigaciones están en pleno desarrollo. Un ejemplo se puede ver en la figura 2.11 [26].

Figura 2.11: Red de sensores de control cerebral

2.3. Sensores de aceleración–inclinación

Los sensores inerciales son usados ampliamente para la medida física de velocidad, aceleración, inclinación y vibración. Tienen aplicaciones en variados tipos de industria como pueden ser la automoción, la electrónica de consumo o la asistencia sanitaria. Es por ello que los acelerómetros son cada vez más utilizados para crear soluciones tecnológicas. Hoy en día toda persona que tenga un smartphone tiene un acelerómetro en el bolsillo que le permite realizar distintas tareas con aplicaciones dedicadas.

Según el ámbito en el que se vaya a utilizar el acelerómetro, se van a requerir diferentes especificaciones de diseño para cumplir con el cometido. Cada actividad tiene unos rangos de aceleración y de frecuencia propios, y por tanto se tienen que desarrollar distintos tipos de sensor que recojan esos requerimientos [27].

Los tipos de acelerómetros existentes son [28]:

- Acelerómetros mecánicos

  Disponen de una masa inercial, resortes elásticos y galgas extensiométricas que son las que miden los cambios de deformación producidos por la aceleración a la que es sometida el dispositivo. Se incluyen sistemas de amortiguación para evitar la oscilación propia. Las galgas extensiométricas hacen de puente entre la carcasa y la masa inercial, la aceleración produce una deformación física en la galga y esto hace que haya una variación de corriente en un puente de Wheatstone. La deformación es directamente proporcional a la aceleración aplicada al acelerómetro.

- Acelerómetros piezoeléctricos

  Su funcionamiento se basa en el efecto piezoeléctrico. Los cristales piezoeléctricos tienen la propiedad de que al ser sometidos a una fuerza producen una corriente eléctrica porque hay una variación en su estructura cristalina. Poniendo un cristal entre la carcasa y una masa inercial, cuando haya una aceleración se producirá una corriente al ejercer la masa una fuerza sobre el cristal. Midiendo esa corriente se puede calcular la aceleración.

- Acelerómetros piezorresistivos

  En este tipo de acelerómetros la fuerza que ejerce la masa inercial es sobre un sustrato, el cual varía su resistencia eléctrica, y al formar parte de un circuito con puente de Wheatstone, se puede medir la variación de la intensidad de la corriente. Una diferencia con la tecnología piezoeléctrica es que este tipo de sensores pueden medir aceleraciones de hasta 0 Hz de frecuencia.

- Acelerómetros capacitivos

  Modifican la posición relativa de las placas de un condensador cuando está sometido a una aceleración, variando así su capacidad. El elemento que conecta la masa inercial con la carcasa es un condensador. Una de las paredes es fija pegada a la carcasa, y la otra pared pegada a la masa inercial, es móvil. Así, cuando hay una aceleración la masa presiona el condensador variando el grosor entre pared y pared, y por tanto, variando la capacidad. Midiendo esa capacidad se puede calcular la aceleración.

- Acelerómetros térmicos

  Este tipo de acelerómetro está basado en la convección termal. En un sustrato de silicio se hace un hueco con una resistencia que hace de calentador, con dos termopares en los extremos. Con esa estructura se consigue formar una cavidad de aire caliente, una burbuja, sobre los termopares. La pequeña burbuja de aire caliente es el elemento móvil, que está herméticamente sellada dentro de una cavidad del sensor. Cuando se aplica una aceleración la burbuja se mueve de una forma análoga. Ese movimiento produce un cambio de temperatura que es detectado por los termopares y produce un voltaje.

- Acelerómetros microelectromecánicos (MEMS)[1]

  Este tipo de acelerómetros tienen como característica principal tener unas dimensiones muy pequeñas, siendo sistemas microelectrónicos, dejando atrás la escala macroscópica que se considera para dimensiones mayores de 0.5 cm. En cuanto a la tecnología básica hay principalmente tres tipos

[1]MEMS: Micro–Electro–Mechanical–System

de acelerómetros MEMS: los capacitivos de silicio, los piezorresistivos y los térmicos. Sin duda, son los capacitivos los que dominan el mercado.

Como cualquier acelerómetro capacitivo, un acelerómetro MEMS capacitivo se basa en la variación de la capacidad del condensador explicada anteriormente. La dificultad está en aplicar esa tecnología en dimensiones microscópicas. Además, un acelerómetro mide la aceleración respecto de un eje concreto, y sin embargo, los dispositivos suelen medir dos o tres ejes simultáneos por lo que se tienen que integrar hasta tres acelerómetros en un mismo componente, aumentando la dificultad. En la figura 2.12 se muestra una oblea para un acelerómetro de 3 ejes ortogonales, y se puede ver cómo orientando dos acelerómetros en el plano se pueden medir los ejes x e y, pero para el eje z, que queda fuera del plano de la oblea, la tecnología tiene que ser distinta [29].

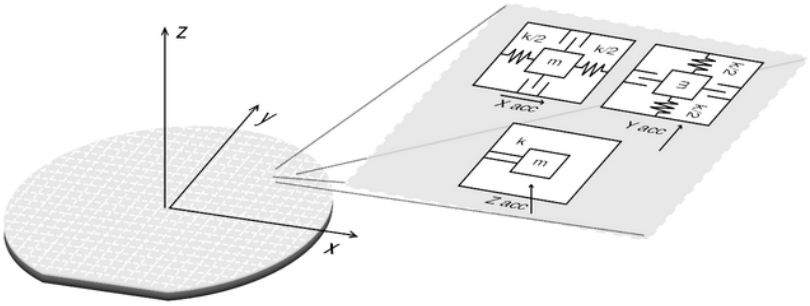


Figura 2.12: Disposición esquemática de los microcondensadores en cada eje de una oblea de silicio

Para seleccionar el acelerómetro adecuado hay que tener en cuenta el rango de aceleraciones en las que se va a trabajar, la frecuencia o ancho de banda necesario, así como parámetros de sensibilidad/resolución y coste, entre otros. En la tabla 2.1 se puede ver un resumen de las características principales de los distintos tipos de acelerómetros [30].

En la interfaz de este proyecto se ha elegido el tipo de acelerómetro MEMS. En primer lugar por el tamaño. Este tipo de acelerómetros son sensiblemente más pequeños que los demás y es objetivo del proyecto diseñar un dispositivo de pequeñas dimensiones. Además la frecuencia de medida en los movimientos y la aceleración en g's de los mismos que se requieren van a ser bajas. Como la frecuencia, considerando el movimiento del brazo, será de menos de 10 Hz, se descarta el tipo piezoeléctrico, y la lentitud y alto coste del tipo mecánico hace que éste se descarte también. Los tipos de acelerómetros restantes están incluidos en el tipo MEMS en cuanto a tecnologías, por tanto se reafirma la decisión de escoger este tipo de acelerómetro. En el capítulo 3 se describe la selección del componente concreto.

Tabla 2.1: Características principales de cada tipo de acelerómetro

Tipo de acelerómetro	Margen de medida	Ancho de Banda (Hz)	Ventajas e inconvenientes	Aplicaciones
Micromecánico	De 1,5 a 250g	De 0,1 a 1500	- Alta sensibilidad - Coste medio - Uso sencillo - Bajas temperaturas	- Impacto - ABS - Airbag - Automoción
Piezo-eléctricos	De 0 a 2000g	De 10 a 20000	- Sensibilidad media - Uso complejo - Bajas temperaturas - No funcionan en continua	- Vibración - Impacto - Uso industrial
Piezo-resistivos	De 0 a 2000g	De 0 a 10000	- Respuesta en continua y alterna - Prestaciones medias - Bajo coste	- Vibración - Impacto - Automoción
Capacitivos	De 0 a 1000g	De 0 a 2000	- Funciona en continua - Bajo ruido - Baja potencia - Excelentes características	- Uso general - Uso industrial
Mecánicos	De 0 a 200g	De 0 a 1000	- Alta precisión en continua - Lentos - Alto coste	- Navegación inercial - Guía de misiles - Herramientas - Nivelación

2.4. Tecnologías inalámbricas. Bluetooth

La escena inalámbrica de corto alcance está representada por cuatro protocolos principales de comunicación: Bluetooth (estándar IEEE 802.15.1), Ultra-wideband UWB (estándar IEEE 802.15.3), ZigBee (estándar IEEE 802.15.4) y Wi-Fi (estándar IEEE 802.11). Cada una de estas tecnologías está originalmente indicada para un tipo determinado de aplicación genérica [31]. Bluetooth está concebido para teclados y ratones de ordenador sin cables, así como para manos libres. UWB está orientado a aparatos multimedia de alto ancho de banda. ZigBee se diseñó para una monitorización de redes inalámbricas fiable y control de redes. Wi-Fi está dirigido a conexiones ordenador–ordenador para sustitución de cableado. Sin embargo, la intención original se queda pequeña para todas estas tecnologías y se pueden utilizar en multitud de desarrollos tecnológicos diferentes. Por ello es importante saber las características de cada una de ellas y así elegir la adecuada para la aplicación que corresponda. A continuación se describe una breve comparativa entre estas tecnologías.

Sin entrar a explicar la base de funcionamiento de cada tecnología, se van a detallar directamente sus características principales, las cuales se pueden ver en la tabla 2.2.

Se puede observar que las tres tecnologías, Bluetooth, ZigBee y Wi-Fi, operan en la misma frecuencia, 2.4 GHz, aunque estas dos últimas tienen otra frecuencia alternativa. UWB sin embargo abarca un rango de frecuencias muy

Tabla 2.2: Comparativa de las tecnologías inalámbricas Bluetooth, UWB, ZigBee y Wi-Fi

COMPARISON OF THE BLUETOOTH, UWB, ZIGBEE, AND WI-FI PROTOCOLS

Standard	Bluetooth	UWB	ZigBee	Wi-Fi
IEEE spec.	802.15.1	802.15.3a *	802.15.4	802.11a/b/g
Frequency band	2.4 GHz	3.1-10.6 GHz	868/915 MHz; 2.4 GHz	2.4 GHz; 5 GHz
Max signal rate	1 Mb/s	110 Mb/s	250 Kb/s	54 Mb/s
Nominal range	10 m	10 m	10 - 100 m	100 m
Nominal TX power	0 - 10 dBm	-41.3 dBm/MHz	(-25) - 0 dBm	15 - 20 dBm
Number of RF channels	79	(1-15)	1/10; 16	14 (2.4 GHz)
Channel bandwidth	1 MHz	500 MHz - 7.5 GHz	0.3/0.6 MHz; 2 MHz	22 MHz
Modulation type	GFSK	BPSK, QPSK	BPSK (+ ASK), O-QPSK	BPSK, QPSK COFDM, CCK, M-QAM
Spreading	FHSS	DS-UWB, MB-OFDM	DSSS	DSSS, CCK, OFDM
Coexistence mechanism	Adaptive freq. hopping	Adaptive freq. hopping	Dynamic freq. selection	Dynamic freq. selection, transmit power control (802.11h)
Basic cell	Piconet	Piconet	Star	BSS
Extension of the basic cell	Scatternet	Peer-to-peer	Cluster tree, Mesh	ESS
Max number of cell nodes	8	8	> 65000	2007
Encryption	E0 stream cipher	AES block cipher (CTR, counter mode)	AES block cipher (CTR, counter mode)	RC4 stream cipher (WEP), AES block cipher
Authentication	Shared secret	CBC-MAC (CCM)	CBC-MAC (ext. of CCM)	WPA2 (802.11i)
Data protection	16-bit CRC	32-bit CRC	16-bit CRC	32-bit CRC

amplio, que también incluye los 2.4 GHz.

UWB es la tecnología que más velocidad de transmisión de datos máxima ofrece, 110 Mbps, seguida de Wi-Fi con 54 Mbps y siendo ZigBee la que menos, 250 kbps. Bluetooth ofrece 1 Mbps (siendo 3 Mbps en la versión más actual).

El alcance de la señal llega a 100 metros aproximadamente en Wi-Fi y ZigBee. En Bluetooth hay dos clases: clase 1 (hasta 100 metros también) y clase 2 (hasta 20 metros). De todas formas, en general, Bluetooth, UWB y ZigBee están orientados a comunicación WPAN[2] para trabajar en torno a 10 metros, mientras que Wi-Fi está orientado a comunicación WLAN[3] para trabajar en torno a 100 metros.

La potencia de transmisión varía desde los 20 dBm de Wi-Fi hasta los 0 dBm de ZigBee, pasando por los 5 dBm de media de Bluetooth.

Por otro lado, el tiempo de transmisión de los datos es más bajo en Bluetooth que en ZigBee, medido según la cantidad total de datos. En Wi-Fi y UWB esta característica es mucho mejor todavía. Ver figura 2.13.

[2]Wireless Personal Area Network
[3]Wireless Local Area Network

En la figura 2.14 se observa que la eficiencia de la codificación de datos es mejor en Bluetooth que en las otras tecnologías para cantidades de datos por debajo de 10^3 bytes.

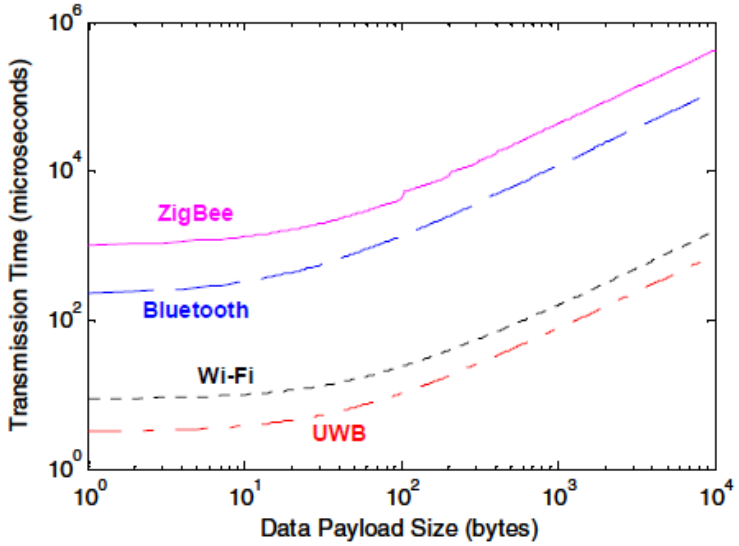


Figura 2.13: Comparativa tiempo de transmisión versus tamaño de datos

El consumo de potencia es muchísimo mayor en Wi-Fi y UWB, mientras que Bluetooth y ZigBee tienen un consumo similar siendo bajo en ambas, con ZigBee ligeramente más ahorradora, sobre todo en transmisión (TX). Ver figura 2.15.

Sin embargo, el consumo de energía normalizado, que es la energía en julios consumida por cada bit transmitido, establece que ZigBee consume aproximadamente el doble que Bluetooth, siendo Wi-Fi y UWB mucho mejores en este apartado. Esto quiere decir que a mayor cantidad de datos transmitidos más consumo tiene ZigBee respecto de Bluetooth, aspecto que tendería a igualarse si los datos transmitidos son pocos. Ver figura 2.16.

Para la aplicación de este proyecto cualquiera de los valores anteriores de potencia de transmisión, alcance y ratio de datos son satisfactorios, ya que que el dispositivo va a estar a pocos metros del ordenador en la misma habitación, se va a transmitir poco volumen de información simultáneo y con una potencia mínima de transmisión se van a poder comunicar los módulos inalámbricos. Por tanto, la característica distintiva es el consumo, y en este apartado el consumo de potencia penaliza mucho a Wi-Fi y UWB, por lo que se descartan. Hay que señalar que el dispositivo será autónomo en cuanto a alimentación y debe tener la máxima autonomía posible, y por tanto requiere el menor consumo. En cuanto al consumo de energía normalizado, ZigBee tiene peor rendimiento que Bluetooth. Recordar que Bluetooth también es la tecnología que mejor eficiencia de codificación de datos tiene.

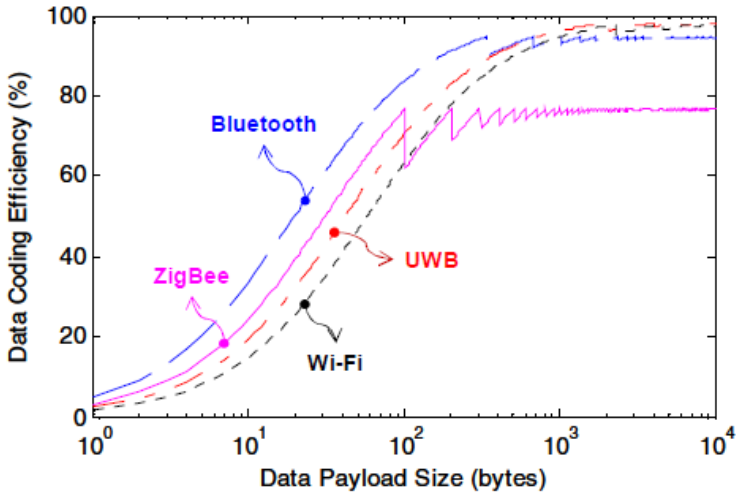


Figura 2.14: Comparativa eficiencia de codificación de datos versus tamaño de datos

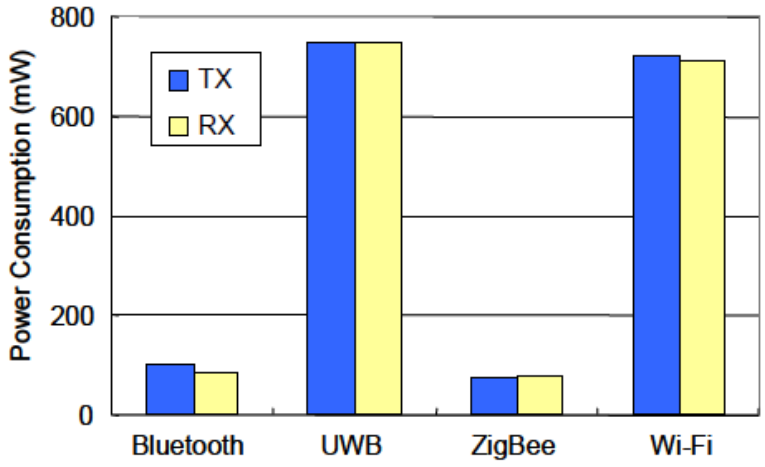


Figura 2.15: Comparativa de consumo de potencia

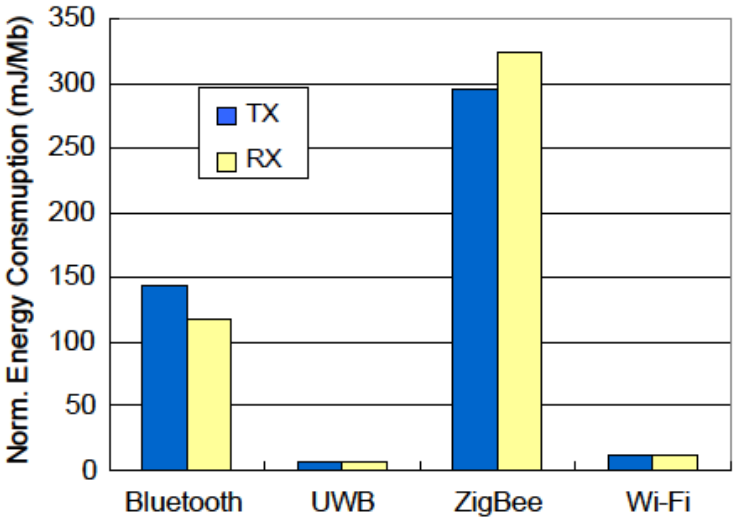


Figura 2.16: Comparativa de consumo de energía normalizado

Hasta ahora la conclusión favorece a Bluetooth respecto de ZigBee. Señalar otros dos aspectos fundamentales en la comparativa, el tamaño y el precio. Aquí hay que buscar y analizar componentes electrónicos concretos. En la tabla 2.3 se comparan los módulos inalámbricos más pequeños que se han encontrado para cada tecnología. La lectura es que aunque hay un módulo ZigBee que consume menos que el módulo Bluetooth, por otro lado su precio es mayor.

Tabla 2.3: Comparativa de módulos inalámbricos concretos

	RN-42 (Bluetooth)	XBEE (Zigbee)	EASYBEE (Zigbee)	WIFLY (Wifi)
Consumo TX (mA)	45	45	18	140
Precio (euros)	18	20	30	50
Tamaño (mm)	25.8 x 13.4 x 2	27.6 x 24.4 x 3	43.5 x 20 x 4	37 x 20 x 3.5
Potencia (dBm)	4	0	0	18
Alcance (m)	20	30	120	100
Ratio Datos (kbps)	240-3000	250	250	1000

Además hay que tener en cuenta una última característica importante a la hora de elegir una tecnología para la interfaz: la universalidad. Esto es, la extensión de la tecnología en todo el mundo, tanto en entornos domésticos como académicos e industriales, así como la facilidad de uso y configuración. Bluetooth es sin duda una tecnología más fácil de encontrar, más fácil de configurar y que

se puede instalar en cualquier ordenador.

Teniendo en cuenta el uso y aplicación que va a tener la interfaz, y teniendo en cuenta toda la comparativa descrita, se selecciona la tecnología Bluetooth.

Bluetooth

En 1994 la empresa sueca de telecomunicaciones Ericsson comenzó un estudio para buscar alternativas al cable que pudieran conectar sus teléfonos móviles con otros accesorios. El estudio se centró en el uso de señales de radio, puesto que no son direccionales, a diferencia de las señales infrarrojas que se solían utilizar en los dispositivos a distancia. A raíz de este estudio nació la especificación Bluetooth. El nombre proviene del rey vikingo Harald Blatand (en inglés, Harold Bluetooth), que fue capaz de unificar y controlar Dinamarca y Noruega. Y la intención de la nueva tecnología era unificar las industrias de la telecomunicación y la informática [32].

Para promocionar y definir la especificación Bluetooth, un grupo de multinacionales decidieron trabajar juntas en el año 1998. Este grupo de promotores se hace llamar el grupo de interés especial Bluetooth (the Bluetooth SIG), y lo conformaron inicialmente Ericsson, Intel, IBM, Toshiba y Nokia, al que un año después se unieron Microsoft, Lucent, 3COM y Motorola. La versión 1.0 de Bluetooth fue publicada en julio de 1999 en la página web `http://www.bluetooth.com`.

Hay una frase que resume muy bien qué es Bluetooth. Es una especificación abierta que utiliza un enlace radio de corto alcance en sustitución de cables para comunicar dispositivos de voz y datos de forma simple y barata, pudiendo ser utilizada en cualquier parte del mundo [33].

Bluetooth se puede clasificar en 3 clases diferentes, atendiendo a la potencia máxima de salida, como se puede ver en la tabla 2.4. La potencia determina el alcance máximo al que puede llegar el dispositivo.

Otras características de la tecnología son:

- Frecuencia. Opera en la banda de frecuencia de 2.4 GHz, no precisando de ninguna licencia.
- Velocidad de transmisión. Dependiendo de la versión de Bluetooth se alcanzan distintas tasas de transferencia de datos. En la versión 1.2 esa tasa es de 1 Mbps, en la versión 2.0 es 3 Mbps, y en las versiones Bluetooth 3.0 y 4.0 se alcanzan los 24 Mbps.

Tabla 2.4: Clases 1, 2 y 3 de Bluetooth

Clase	Potencia máxima [mW]	Potencia máxima [dBm]	Alcance máximo [metros]
1	100	20	100
2	2,5	4	10
3	1	0	1

- Bajo consumo de energía. La clase 2 es la más extendida y sus 2.5 mW de potencia significan un muy bajo consumo de energía.
- Bajo precio. El coste de los integrados es menor a 3 dólares (US$).
- Omnidireccional. Al transmitir en todas direcciones no es necesario que los dispositivos a enlazar estén en la misma línea de visión.

La especificación Bluetooth se basa en una pila de protocolos, que permite a los dispositivos localizarse, conectarse e intercambiar datos entre ellos, y permite que puedan ejecutar aplicaciones interactivas. La pila de protocolos completa se puede ver en la figura 2.17 [34]. Destacar los protocolos L2CAP y RFCOMM. RFCOMM funciona sobre el protocolo L2CAP, y es el encargado de emular el puerto serie RS-232, actuando de puerto virtual, y su importancia es enorme puesto que es el sustituto inalámbrico de la comunicación serie que hasta entonces sólo era implementada a través de cable.

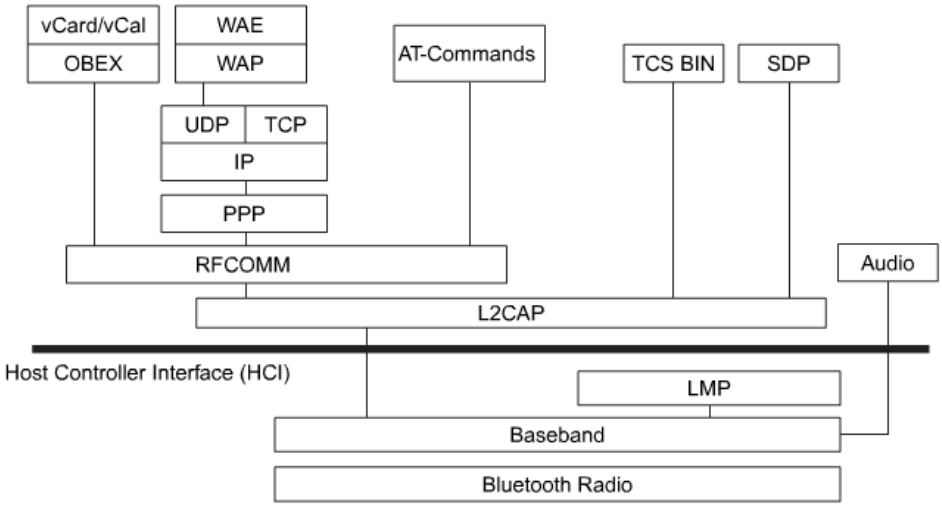


Figura 2.17: Pila de protocolos Bluetooth

A continuación se van a presentar algunos módulos electrónicos Bluetooth utilizados en ingeniería.

- Bluetooth KC11 de KCWirefree (ver figura 2.18) [35]
  - Bluetooth 1.2. Velocidad hasta 921.6 kbps.
  - Clase 1. Potencia de salida 18 dBm. Alcance hasta 100 m.
  - Voltaje de entrada: 2.7 – 3.6 V.
  - Consumo máximo: 210 mA.
  - Tamaño: 15 mm x 43 mm x 2.0 mm.

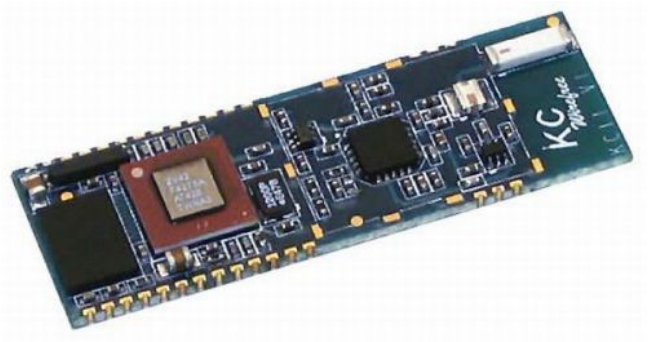


Figura 2.18: Módulo Bluetooth KC11

- Bluetooth WT12 de Bluegiga (ver figura 2.19) [36]
  - Bluetooth 2.1. Velocidad hasta 3 Mbps.
  - Clase 2. Potencia de salida 3 dBm. Alcance hasta 30 m.
  - Voltaje de entrada: 2.7 – 3.6 V.
  - Consumo máximo: 70 mA.
  - Tamaño: 14 mm x 25.5 mm x 2.0 mm.

Figura 2.19: Módulo Bluetooth WT12

- Bluetooth HC-05 de Wavesen (ver figura 2.20) [37]

  - Bluetooth 2.0. Velocidad hasta 2 Mbps.
  - Clase 2. Potencia de salida 4 dBm. Alcance no especificado.
  - Voltaje de entrada: 3.1 – 4.2 V.
  - Consumo máximo: 40 mA.
  - Tamaño: 13 mm x 27 mm x 2.0 mm.

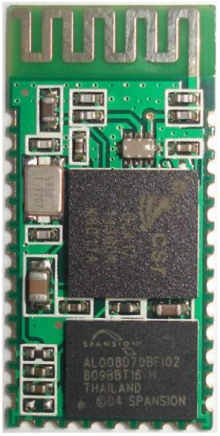


Figura 2.20: Módulo Bluetooth HC-05

- Bluetooth RN42 de Roving Networks (ver hoja de características en el Apéndice E y figura 3.12 del capítulo 3)

  - Bluetooth 2.1. Velocidad hasta 3 Mbps.
  - Clase 2. Potencia de salida 4 dBm. Alcance hasta 20 m.
  - Voltaje de entrada: 3 – 3.6 V.
  - Consumo máximo: 50 mA.
  - Tamaño: 13.4 mm x 25.8 mm x 2.0 mm.

Aunque la selección del componente concreto de la interfaz del proyecto, que es el módulo RN42, se explica en el capítulo 3, esta comparativa sirve para ver las características de otros módulos Bluetooth, y constatar que los dos primeros tienen un consumo demasiado alto para los objetivos propuestos. Por otro lado, y aunque el módulo HC-05 es muy parecido al RN42, el tamaño decide, puesto que es un objetivo más prioritario que el consumo en este caso.

Capítulo 3

Descripción general

El dispositivo que se presenta en este documento es una interfaz hombre–máquina (HMI) inalámbrica diseñada para facilitar el control de sistemas adscritos al ámbito de la robótica asistencial. La función principal del dispositivo es detectar los movimientos en el espacio de la persona, y trasladar esa información de forma inalámbrica a un ordenador. En un futuro desarrollo la información recibida en el ordenador sería trasladada al robot. Ver figura 3.1.

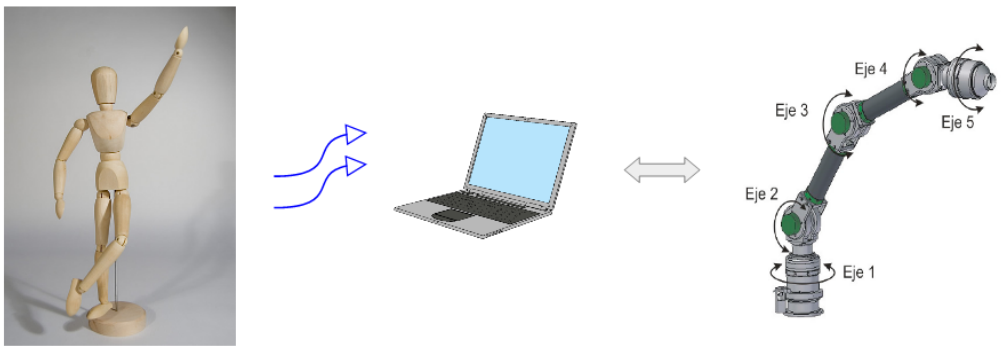


Figura 3.1: Funcionalidad de la interfaz hombre–máquina (HMI)

El fundamento de la interfaz sigue la siguiente secuencia de acciones: recogida de datos – tratamiento de datos – envío de datos – recepción de datos, como se puede ver en la figura 3.2. Por tanto, se trata de encontrar la solución tecnológica que permita medir unos datos físicos determinados, tratarlos y enviarlos a un ordenador.

Esta secuencia se traduce en el esquema funcional de la figura 3.3: acelerómetro – microcontrolador – módulo inalámbrico – ordenador (PC).

Figura 3.2: Secuencia de acciones de la interfaz

El dispositivo integra en una misma placa de circuito impreso (PCB) el acelerómetro, el microcontrolador y el módulo inalámbrico, todo ello incluido dentro de un soporte plástico que le da una estructura mecánica. Es acoplado a la persona que ejerce el control en una parte móvil de su cuerpo, normalmente un brazo, de tal manera que queda unido de forma solidaria. Así, el movimiento del brazo es seguido por el dispositivo.

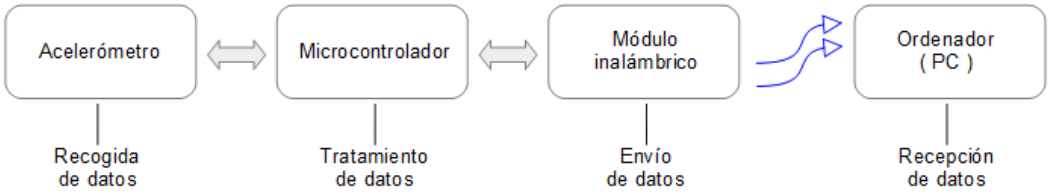


Figura 3.3: Esquema funcional general

El acelerómetro detecta movimiento en los tres ejes del espacio, x, y, z, por medio de la inclinación en cada uno de los ejes respecto de una posición de inicio o reposo. Esa inclinación es medida y cuantificada, y enviada al microcontrolador. El microcontrolador es el encargado de gestionar los datos recibidos del acelerómetro. Mediante una programación adecuada se efectúa el tratamiento de los datos para interpretar el movimiento de cada eje, y a su vez se transforman dichos datos para ser enviados al módulo inalámbrico. Este módulo se encarga de enviar los datos sin necesidad de cable hacia el ordenador, el cual los recibe e interpreta mediante la adecuada programación y ejecución de un programa cliente.

Para poder llevar a cabo de forma correcta lo anterior es clave la correcta programación tanto del microcontrolador como del ordenador. Por ello, una parte fundamental del dispositivo es el software a programar, que será el encargado de aprovechar de forma óptima el hardware utilizado.

Además, el dispositivo es autónomo, puesto que lleva incorporadas baterías (pilas), evitando así la dependencia de alimentación externa que conllevaría cables, los cuales son contrarios al objetivo pretendido de dimensiones pequeñas, poco peso y manejabilidad.

La estructura mecánica sobre la que va insertada la PCB es un soporte de plástico de diseño propio conformado por una impresora 3D, al cual van añadidas unas ranuras para incorporar una correa a modo de reloj en el caso de que el dispositivo se acople en el brazo.

Por tanto, la finalidad de la interfaz es servir de intermediaria entre la persona y el ordenador para que los movimientos reales efectuados por el primero puedan ser entendidos por el segundo.

La aplicación directa para el movimiento del robot ASIBOT conlleva una programación adicional para comunicar el ordenador con el robot, que será comentada en el capítulo de desarrollos futuros. En este apartado reseñar que para una funcionalidad completa de la interfaz respecto del robot hay que añadir un sistema de pulsadores complementarios no desarrollados en este proyecto. Sin embargo, sí que se describe una detección de interrupción en el dispositivo que hace la función de pulsador, dando al sistema interfaz–robot una funcionalidad intermedia. Este aspecto se describe con detalle más adelante.

3.1. Descripción funcional

Según lo descrito en el apartado anterior el dispositivo tiene tres funciones principales, recogida de datos, tratamiento de datos y envío de datos. Externo al dispositivo en sí, pero siendo parte de la interfaz, queda la función de recepción de datos por parte del ordenador. Estas funciones las llevarán a cabo, respectivamente, el acelerómetro, el microcontrolador (PIC), el módulo inalámbrico Bluetooth y el ordenador (PC); figura 3.4.

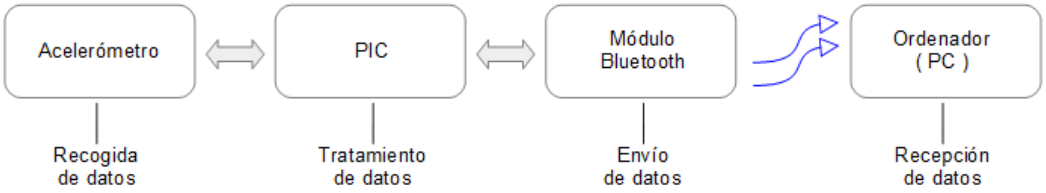


Figura 3.4: Esquema funcional particular

La función de recogida de datos es efectuada por el acelerómetro. Éste mide la magnitud física de inclinación respecto de una posición inicial, en cada uno de los 3 ejes espaciales, x, y, z, y transmite estos datos al microcontrolador a través de una conexión serie. Además de los valores recogidos a tiempo real en cada eje, se habilita la capacidad de detección de una aceleración grande en el eje z a modo de interrupción (tap).

El microcontrolador (PIC) es el cerebro del dispositivo. Mediante un programa escrito en lenguaje C es capaz de recibir los datos procedentes del acelerómetro mediante una conexión serie, transformarlos e interpretarlos de manera adecuada, para después transmitirlos hacia el módulo Bluetooth mediante otra conexión serie distinta. Está programado también para detectar una interrupción proveniente del acelerómetro si se produce. Dispone de un circuito de programación a través del cual puede ser programado de forma externa.

Para enviar los datos de forma inalámbrica se utiliza un módulo Bluetooth. La justificación de elegir esta tecnología se ha descrito en el capítulo 2. Este módulo recibe por el puerto serie los datos tratados del microcontrolador y los envía por medio de una antena integrada en el propio chip.

Para captar estos datos se instala en un ordenador que disponga de Bluetooth un programa cliente que, basado en programación mediante sockets, es capaz de recibir información Bluetooth, tratarla y mostrarla en pantalla (en un futuro desarrollo esos datos se enviarían al robot).

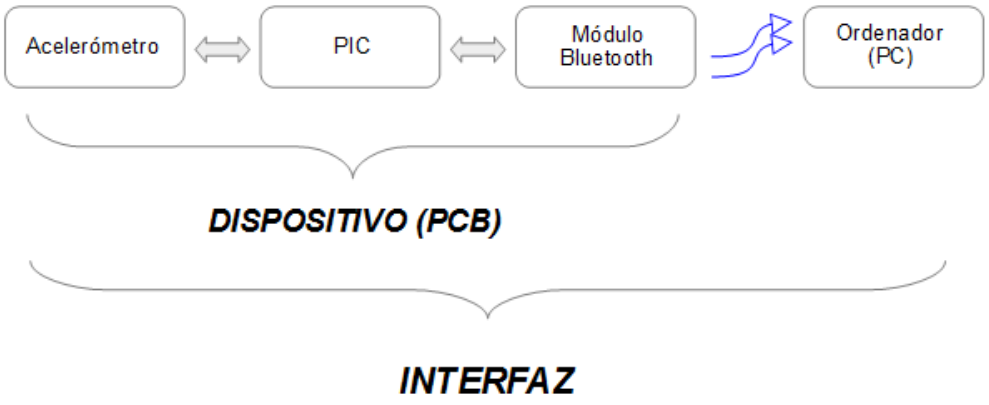


Figura 3.5: Dispositivo vs Interfaz

Por tanto, y como se puede ver en la figura 3.5, queda definido como dispositivo propiamente dicho el conjunto de acelerómetro–PIC–Bluetooth, integrados dentro de la misma PCB, mientras que se define a la interfaz como la unión del dispositivo junto con el programa cliente del ordenador que recibe los datos, ya que es todo el conjunto el que completa la misión de comunicar persona y máquina.

3.2. Especificaciones de diseño

Para acometer la funcionalidad que se pretende para la interfaz se buscan las características técnicas más apropiadas que conjuguen igualmente con los objetivos, que son entre otros, dimensiones lo más pequeñas posibles, peso mínimo y consumo mínimo.

Hay varias especificaciones generales a tener en cuenta. Se quiere unificar la alimentación del dispositivo, es decir, se va a buscar un nivel de voltaje común para alimentar todos los componentes del circuito, con el objetivo de eliminar circuitos de acondicionamiento particulares que harían aumentar el número de componentes total, y por tanto, las dimensiones, el peso y el consumo del conjunto. En ese sentido, y por la misma razón, otra especificación de diseño principal es la elección de componentes digitales, y no analógicos.

En primer lugar, el acelerómetro que se necesita es un sensor de 3 ejes digital; figura 3.6. El movimiento de la persona se efectúa dentro de los tres ejes espaciales, x, y, z, así que el número de ejes sobre los que debe medir el sensor es de 3. Se busca un acelerómetro digital, y no analógico, principalmente porque reduce la necesidad de circuitos auxiliares de acondicionamiento de señal, dejando al mínimo el número de componentes electrónicos asociados. Con un sensor analógico sería necesario acondicionar la señal de salida del mismo, siendo necesarios muchos componentes extra. Como ya se indicó anteriormente, se busca reducir al mínimo las dimensiones, el peso y el consumo, así que la opción ideal es escoger un sensor digital. Por otro lado, el sensor debe tener un puerto de comunicación serie digital, para poder enviar la información medida al microcontrolador. También es necesario que soporte interrupciones, ya que esta característica será implementada en el diseño.

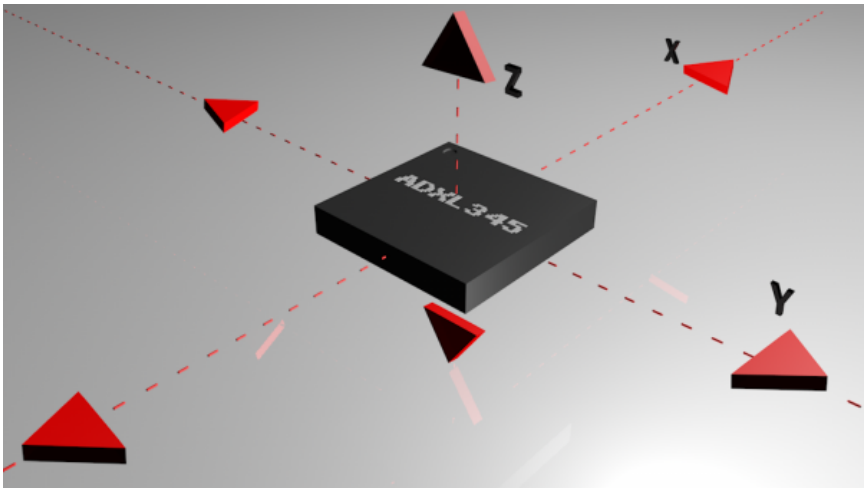


Figura 3.6: Sensor genérico de 3 ejes

El microcontrolador (digital) tiene que tener las siguientes características: Un procesador de 8 bits. Un ancho del bus de datos de 8 bits es suficiente para el tratamiento de los datos procedentes del acelerómetro. Dos puertos de comunicación serie. SPI y USART. Uno para comunicarse con el acelerómetro y el otro para comunicar con el módulo Bluetooth. Posibilidad de ser programado y reprogramado una vez instalado (In-Circuit Programming). Al menos 3 pines de puertos genéricos de entrada/salida libres, adicionales a los ocupados por los puertos serie y los pines de programación. Capacidad de soportar al menos 2 interrupciones. Además, debe de tener una memoria de programa y una velocidad de funcionamiento suficientes, no se calcula un requisito concreto de antemano. Se puede ver un microcontrolador genérico de encapsulado DIP en la figura 3.7.

Figura 3.7: Microcontrolador genérico encapsulado DIP

El módulo Bluetooth (figura 3.8) debe de tener un puerto de comunicación serie para recibir los datos del microcontrolador (USART). El alcance de la señal Bluetooth no hace falta que sea superior a 10 metros puesto que el dispositivo se va a comunicar con el ordenador receptor en la misma habitación. Es importante que la antena de comunicación sea lo más pequeña posible y a ser posible, integrada en el chip del módulo inalámbrico. Por supuesto, este componente debe de ser digital y ocupar en conjunto lo menos posible.

En cuanto a la alimentación del dispositivo, se hará mediante baterías para que el conjunto sea autónomo e independiente de alimentación exterior alguna. Por el hecho de conseguir el menor tamaño posible en el conjunto, las baterías serán pilas botón. Deberá conseguirse un equilibrio entre tamaño y autonomía. Como la autonomía depende del consumo de la carga que componen los componentes finales seleccionados, se buscará en las pilas la máxima capacidad posible respecto de un tamaño razonable.

Para unificar el nivel de voltaje en la alimentación, se va a elegir el valor de 3.3 V. La especificación de diseño previa es unificar a un valor común la alimentación, y aunque este valor es encontrado al realizar la selección de componentes explicada en el siguiente apartado, se va a explicar aquí. Una vez encontrados los

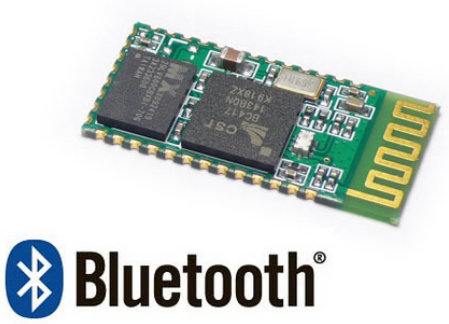


Figura 3.8: Módulo Bluetooth genérico

componentes principales del dispositivo, que son el acelerómetro, el microcontrolador y el módulo Bluetooth, se llega a un rango de valores de alimentación para cada uno, que se pueden ver en la tabla 3.1.

Tabla 3.1: Rango de voltaje de alimentación de los componentes principales

Componente	Rango Voltaje de Alimentación [V]	Voltaje típico [V]
Acelerómetro digital ADXL345	2 – 3,6	2,5
Microcontrolador PIC16LF876A	2 – 5,5	-
Módulo Bluetooth RN42	3 – 3,6	3,3

Observando estos rangos de valores, y teniendo en cuenta que el valor típico de funcionamiento del módulo Bluetooth es de 3.3 V, se elige este valor, como valor de alimentación común para todo el circuito.

Para obtener este valor de alimentación se va a necesitar un regulador de tensión cuyo valor de salida sea precisamente ese, 3.3 V. Este regulador debe ofrecer también una eficiencia máxima para entregar una señal lo más estable posible. Además es importante que necesite de los menos pasivos auxiliares posible. Y por último, será imprescindible que tenga un rango de valores de entrada que abarque el voltaje total que entregarán las baterías.

Para visualizar esto último, y suponiendo el voltaje más común para una pila botón, que es de 1.5 V, hacen falta 3 pilas en serie para superar el voltaje deseado de 3.3V, llegando a un total de 4.5 V. Este voltaje será el voltaje de entrada del regulador.

Todo este proceso de unificación de voltaje en la alimentación se puede ver en la figura 3.9.

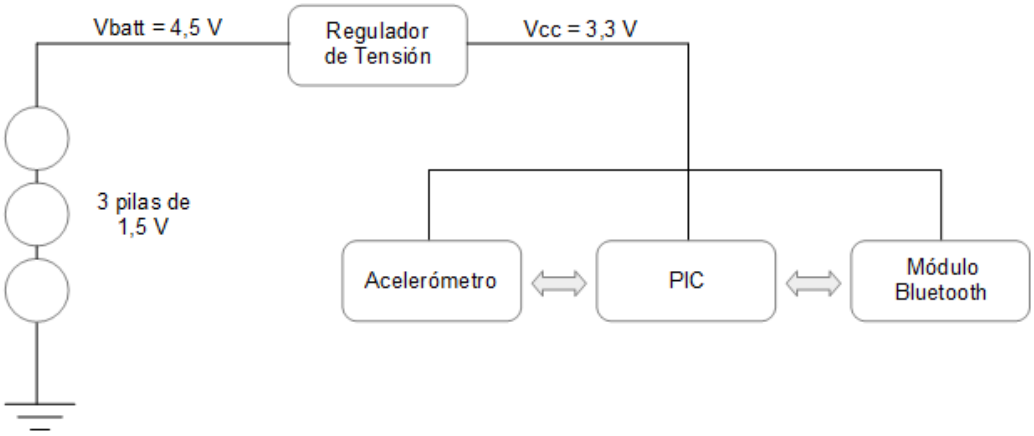


Figura 3.9: Unificación de la alimentación

3.3. Selección de componentes

Los componentes exactos seleccionados se pueden ver en el Anexo A, tanto para la placa experimental que sirve de entorno de pruebas, como para la placa definitiva. La mayoría de los componentes son los mismos en ambos listados en cuanto a funcionalidad, pero cambiando de forma sustancial en cuanto a tamaño y tipo de montaje principalmente. A continuación se va a explicar con detalle la elección concreta de cada componente [38].

- **Acelerómetro digital ADXL345**

  Del fabricante Analog Devices, es un acelerómetro digital de 3 ejes. Dispone de comunicación serie, en los protocolos SPI e I^2C, así como 2 pines para interrupciones, que son las especificaciones previas que se requerían. Tiene una resolución de hasta 13 bits, midiendo hasta +/- 16g de aceleración. Esta resolución le permite detectar cambios de inclinación de menos de 1 grado. La información de salida es codificada en 16 bits en complemento a 2. El rango de voltaje de entrada es de 2 a 3.6 V. Su consumo es muy pequeño, en torno a 30 μA en modo de medición. Además tiene un tamaño muy pequeño también, siendo 5 mm su dimensión mayor, lo cual se puede apreciar en la figura 3.10. Estas características principales se resumen en la tabla 3.2.

  Para seleccionar este componente en primer lugar se ha filtrado para buscar entre todos los acelerómetros digitales de 3 ejes. El siguiente requisito era asegurarse de que dispone de comunicación serie y una resolución alta. A mayor resolución, más sensible es el sensor y puede detectar cambios más pequeños en la inclinación, dando lugar a una mayor precisión en el seguimiento del movimiento. También era necesario que funcionara a

Tabla 3.2: Resumen características principales del ADXL345

ADXL345

Característica	Valor
Fabricante	Analog Devices
Analógico / Digital	Digital
Número de ejes	3
Resolución	Hasta 13 bits
Formato de salida	16 bits en complemento a 2
Comunicación serie	SPI e I²C
Rango voltaje de entrada	2 – 3,6 V
Consumo	30 uA (typ)
Dimensiones	3 x 5 x 1 mm

3.3 V y que su consumo y dimensiones fueran lo menores posible. Por supuesto el tipo de montaje debe de ser superficial.

- **Microcontrolador PIC16LF876A**

  Del fabricante Microchip, este microcontrolador tiene un procesador de 8 bits, una frecuencia de reloj de hasta 10 MHz y una memoria de programa Flash de 8K x 14bit palabras. Dispone de 28 pines en total, con 3 puertos genéricos A,B y C de entrada/salida (22 pines). Tiene comunicación serie síncrona, SPI e I^2C, y comunicación serie asíncrona, USART. Soporta hasta 14 interrupciones. Además, se puede programar tanto antes como después de instalarlo a través de 2 pines con el sistema In-Circuit Serial Programming de Microchip. Este modelo de la familia PIC16, es el LF876 (L de low, bajo, en inglés), que permite operar a voltajes menores que el modelo F876. Este rango de voltaje de funcionamiento va de 2 a 5.5 V. Su consumo alcanza un máximo de 2 mA operando a una frecuencia de 4 MHz, siendo el consumo típico de 0.6 mA. La dimensión más desfavorable en el modelo de montaje superficial es de 10.2 mm. La tabla 3.3 es un resumen de las características principales. Se puede ver el aspecto del PIC en la figura 3.11.

  Es condición imprescindible que el voltaje de funcionamiento sea de 3.3 V por lo que había que encontrar un microcontrolador que se ajustara a

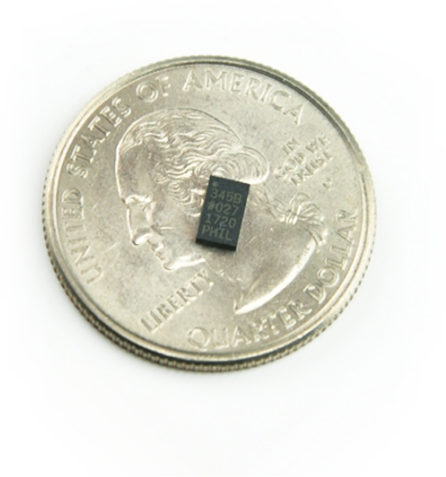


Figura 3.10: Acelerómetro ADXL345 (con referencia de tamaño)

Tabla 3.3: Resumen características principales del PIC16LF876A

PIC16LF876A

Característica	Valor
Fabricante	Microchip
Procesador	8 bits
Frecuencia de reloj	Hasta 10 MHz
Memoria de programa	8 K x 14bits(words)
Número de pines	28
Número de puertos I/O	3
Número de interrupciones	14
Comunicación serie	SPI / I^2C y USART
Rango voltaje de entrada	2 – 5,5 V
Consumo	0,6 mA (typ)
Dimensiones	5,3 x 10,2 x 1,75 mm

esa condición. Muchos microcontroladores operan a voltajes mayores, por lo que era preciso encontrar un modelo de bajo voltaje como es el caso. Además, era también imprescindible que se pudiera programar y reprogramar una vez instalado, y toda la familia de microcontroladores PIC de Microchip tienen un entorno de programación llamado MPLAB IDE y un programador llamado MPLAB ICD2, que permite realizar esta tarea de forma sencilla, a través del sistema In-Circuit Serial Programming. A la hora de seleccionar el componente concreto también era obligatorio que dispusiera de 2 puertos de comunicación serie, el PIC elegido cumple la condición al poder operar mediante SPI (o I^2C) y USART de forma independiente. A la vez había que tener en cuenta que una vez descontados los pines usados en ambas comunicaciones serie, debían quedar libres suficientes pines de entrada/salida genéricos para hacer otras conexiones auxiliares. Esto se cumple; ver conexiones del PIC en los esquemáticos del Anexo C. Por lo demás, se cumple el requerimiento de soportar al menos 2 interrupciones.

En cuanto a la memoria de programa se calcula suficiente la capacidad de 8K palabras de 14 bits(instrucciones), para un código de programa de aproximadamente 500 líneas. Una instrucción en lenguaje ensamblador ocupa una palabra de 14 bits. Se va a programar en lenguaje C, y se puede estimar que una línea de código escrita en C ocupa una media de 4 instrucciones en lenguaje ensamblador (este valor varía dependiendo de la complejidad del mandato de la línea de código, pudiendo oscilar entre 1 y 8 instrucciones en las operaciones más habituales). Por ello se calcula en la ecuación 3.1,

$$8K\,\text{instrucciones}\,\frac{1\,\text{línea en C}}{4\,\text{instrucciones}} = 2K\,\text{líneas en C} \tag{3.1}$$

Por tanto, de la ecuación se deduce que en término medio la memoria de programa del microcontrolador puede albergar 2048 líneas de código en lenguaje C, y como se estima escribir un programa de unas 500 líneas, se da por suficiente dicha memoria.

- **Módulo Bluetooth RN42**

  Del fabricante Roving Networks, es un chip con antena integrada de tecnología Bluetooth 2.0 de clase 2, por lo que tiene un alcance de hasta 20 metros. Dispone de una interfaz de conexión serie USART con una tasa de transferencia de datos desde 240 kbps hasta 3 Mbps, según modo. Incluye los protocolos Bluetooth RFCOMM y L2CAP. Su velocidad de transmisión puede ser programada desde 1200 bps hasta 921 kbps. Consume hasta 50 mA transmitiendo y su dimensión más grande es de 25.8 mm. El voltaje de alimentación puede ir de 3 a 3.6 V, siendo su voltaje típico 3.3 V. Ver tabla 3.4.

Figura 3.11: Microcontrolador PIC16LF876A

Tabla 3.4: Resumen características principales del RN42
RN42

Característica	Valor
Fabricante	Roving Networks
Protocolos Bluetooth	RFCOMM y L2CAP
Alcance	Hasta 20 metros
Velocidad de transmisión	1200 bps – 921 kbps
Tasa de transferencia serie	240 kbps – 3 Mbps
Comunicación serie	USART
Rango voltaje de entrada	3 – 3,6 V
Consumo	45 mA (typ) / 50 mA (max)
Dimensiones	13,4 x 25,8 x 2 mm

Un requerimiento fundamental era tener las dimensiones más pequeñas posibles, puesto que los módulos emisores de radiofrecuencia suelen ser grandes, sobre todo, por la inclusión de la antena. Este componente es el emisor de Bluetooth más pequeño que se ha podido encontrar en el mercado y tiene la antena integrada en el chip. Su alcance es suficiente para la aplicación que se le va a dar y dispone de un puerto serie USART para comunicarse con el microcontrolador. Sin embargo, en este caso tanto la tasa de transferencia de datos serie como la velocidad de transmisión no son decisivas, puesto que la generación de los datos en el acelerómetro dependen de un movimiento físico, y tales datos nunca van a alcanzar la tasa de transferencia más restrictiva de este módulo. Es conveniente notar que este componente es el más desfavorable de los tres componentes principales vistos hasta ahora, en cuanto a dos de las características claves para cumplir objetivos, el consumo y el tamaño. Por ello, se resalta el esfuerzo realizado a la hora de buscar un módulo Bluetooth que fuera lo más pequeño y con el menor consumo posibles. Se puede ver el RN42 en la figura 3.12.

Figura 3.12: Módulo Bluetooth RN42

- **Regulador de tensión TPS62203**

  Del fabricante Texas Instrument, es un regulador programable de tensión continua-continua de alta eficiencia basado en modulación por ancho de pulsos (PWM). Es un convertidor reductor de tensión (y elevador de intensidad) cuyo rango de voltaje de entrada es de 2.5 a 6 V, y cuyo voltaje de salida es exactamente 3.3 V. Ofrece una corriente de salida de hasta 300 mA. Su eficiencia máxima es del 95 % en condiciones óptimas. Ver tabla 3.5. Requiere de dos condensadores y una bobina externa para funcionar. Es de muy pequeñas dimensiones; figura 3.13.

Tabla 3.5: Resumen características principales del TPS62203
TPS62203

Característica	Valor
Fabricante	Texas Instrument
Rango voltaje de entrada	2,5 – 6 V
Voltaje de salida	3,3 V
Corriente de salida	Hasta 300 mA
Eficiencia	95,00%
Dimensiones	1,6 x 2,9 x 1,15 mm

La especificación de diseño principal para este componente es el ajuste en la tensión de salida a exactamente 3.3 V. Dentro de la familia TPS el modelo 62203, ajusta el voltaje de salida justo a ese valor. Esta señal es muy estable puesto que se alcanza el máximo de la eficiencia, ya que la estimación de corriente consumida por la carga del circuito final es de aproximadamente 50 mA; ver ecuaciones (3.2) y (3.3). Según la hoja de características de este componente (ver Apéndice E), con una corriente de carga de 50 mA y una tensión de entrada de 4.5 V (3 pilas en serie de 1.5 V cada una), se alcanza el máximo de eficiencia, un 95 % aproximadamente.

$$i_{carga} = i_1 + i_2 + i_3 \tag{3.2}$$

$$i_{carga} = 0{,}03 + 0{,}6 + 50\,\text{mA} = 50{,}63\,\text{mA} \tag{3.3}$$

Donde:

i_1 = Corriente aproximada consumida por ADXL345
i_2 = Corriente aproximada consumida por PIC16LF876A
i_3 = Corriente aproximada consumida por RN42

- **Oscilador de Cristal 4 MHz**

  El microcontrolador PIC16LF876A necesita conectar un oscilador externo como referencia para sus ciclos de reloj. Según las hojas de características (ver Apéndice E), este PIC puede tener como máximo una frecuencia de reloj de 10 MHz, y según el modo XT (cristal/resonador) la frecuencia

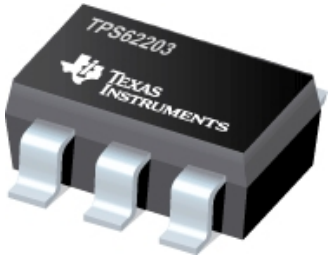


Figura 3.13: Regulador de tensión TPS62203

más alta a escoger es 4 MHz. Escogemos esta frecuencia pues es adecuada para el propósito del PIC.

$$\text{Frecuencia de reloj} = 4 \text{ MHz} = 4 \text{ x } 10^6 \text{ ciclos/seg}$$

$$\frac{1}{4\,\text{MHz}} = 0{,}25 \cdot 10^{-6}\,\text{seg cada ciclo} = 250\,\text{ns cada ciclo reloj} \tag{3.4}$$

$$1 \text{ ciclo instrucción} = 4 \text{ ciclos de reloj}$$

$$250\,\text{ns} \cdot 4 = \frac{1000\,\text{ns}}{\text{ciclo instrucc}} = \frac{1\,\mu\text{s}}{\text{ciclo instrucc}} \tag{3.5}$$

$$\text{Memoria máxima} = 8\text{K instrucciones}$$

$$\frac{1\,\mu\text{s}}{\text{ciclo instrucc}} \cdot 8K\,\text{instrucciones} = 8\,\text{ms} \tag{3.6}$$

Efectivamente, que el PIC tarde 8 ms en ejecutar una vuelta completa al código programado en la memoria de programa suponiendo ésta llena, es aceptable.

Para la placa experimental se elige un oscilador de cristal genérico de 4 MHz con montaje pasante. En la placa definitiva el componente elegido es el oscilador de cristal SG310SCF del fabricante Epson, de montaje superficial (SMD), y frecuencia 4 MHz, que se puede ver en la figura 3.14. El voltaje de entrada típico es de 3.3 V, lo que se ajusta al voltaje unificado de alimentación para todos los componentes del circuito. Además, se buscaba que tuviera las dimensiones más pequeñas posibles.

- **Diodos LED**

  Para el control de estado de funcionamiento del dispositivo se va a conectar un LED rojo desde la línea de alimentación de 3.3 V a masa, con su correspondiente resistencia. El diodo elegido para la placa definitiva es el HSMC-A431-X90M1, del fabricante Avago Technologies. Tipo de montaje superficial. Se buscaba el menor tamaño posible y que tuviera una forma

Figura 3.14: Oscilador de cristal 4 MHz SMD

determinada para acoplar en el diseño final. Las características eléctricas y lumínicas son menos relevantes, aunque adecuadas.

Para el control del estado de transmisión del módulo Bluetooth RN42 se va a conectar un LED verde desde la línea de alimentación de 3.3 V al pin PIO5 del RN42, con su correspondiente resistencia. El diodo elegido para la placa definitiva es el HSMM-A430-X90M2, de Avago Technologies; figura 3.15. Tipo de montaje superficial. Es idéntico en tamaño y forma al LED anterior. El funcionamiento exacto de este LED se explicará más adelante.

Figura 3.15: Diodos LED de montaje superficial

- **Pasivos**

  Los pasivos utilizados en el dispositivo serán explicados con más detalle en el capítulo de Diseño Hardware. En el Apéndice B aparecen los listados detallados de estos componentes tanto para la placa experimental de pruebas como para la placa definitiva. En esta última los pasivos son del fabricante RS, de montaje superficial, y se ha buscado el menor tamaño posible, es decir, la huella más pequeña para cada valor del componente, intentando a la vez unificar las huellas en el menor número posible. Concretamente todos los pasivos seleccionados tienen huella 0402 ó 0603, su aspecto se puede ver en la figura 3.16.

  Los componentes utilizados son los siguientes:

- Condensadores: 10 μF (x1), 4.7 μF (x1), 100 nF (x5).
- Resistencias: 10 kΩ (x1), 1 kΩ (x1), 220 Ω (x2), 10 Ω (x1).
- Inductancia: 10 μH (x1).

Figura 3.16: Pasivos de montaje superficial

- **Conector microUSB hembra**

  Se quiere dotar al dispositivo de la capacidad de ser reprogramado en cualquier momento durante la vida útil del mismo. Para ello se busca un puerto de conexión lo más pequeño posible que se pueda integrar con facilidad en el resto del diseño. La programación del PIC mediante el sistema de In-Circuit Serial Programming necesita de dos líneas de datos y una línea de voltaje de programación. El estándar USB, en sus conectores microUSB, dispone de 5 patillas (figura 3.17), por lo que si sumamos a las 3 líneas que necesitamos, la línea de alimentación del USB y la línea de conexión a masa, se obtienen justo las 5 líneas de las que dispone el conector microUSB.

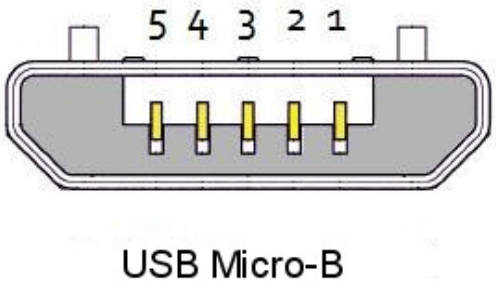


Figura 3.17: Patillaje del conector microUSB

La programación se realiza a través del programador MPLAB ICD2, cuyo terminal de conexión es del tipo RJ11. Montar un conector RJ11 hembra en el dispositivo final se descarta pues es demasiado grande, por lo que

se ha buscado la solución en los conectores USB. Dentro de éstos, el más pequeño, y finalmente seleccionado, es el microUSB. Por ello, a la hora de programar desde el ICD2, ha sido necesario construir un conector personalizado que transforme una conexión RJ11 a una conexión microUSB. Este conector será descrito en el capítulo de Diseño Hardware. Por otro lado, el tipo de montaje del conector final tiene que ser superficial. Así, el componente elegido como conector microUSB hembra es del fabricante TE Connectivity, figura 3.18.

Figura 3.18: Conector microUSB hembra

- **Interruptor Deslizante**

  Para tener el control del encendido o funcionamiento del dispositivo se incluye en el diseño un interruptor o switch, de 2 posiciones, ON/OFF. El componente elegido es el interruptor deslizante MMP121-R del fabricante Knitter Switch; ver figura 3.19. Se ha buscado un componente de montaje superficial con las menores dimensiones posibles, sin embargo, todos los interruptores encontrados eran de tipo pasante, por lo que se ha estudiado muy bien la forma concreta del interruptor a la hora de incluirlo en el diseño final. Al tener conexionado de doble cara en la PCB final, con cara top y cara bottom, los pines pasantes del interruptor se conexionan en la cara bottom, por lo que queda solucionado que no sea un componente de tipo superficial como se pretendía en un principio.

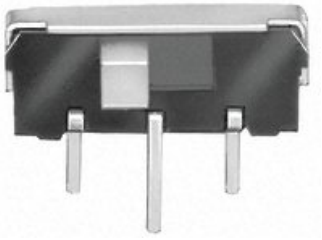


Figura 3.19: Interruptor deslizante

- **Portapilas botón para PCB**

El componente elegido es el portapilas 2996 del fabricante Keystone, con tipo de montaje superficial; ver figura 3.20. Soporta pilas botón de 11.6 mm de diámetro, y específicamente soporta, entre otras, la pila tipo SR44. Se necesitan 3 unidades, tantas como pilas utilizadas en el dispositivo.

Figura 3.20: Portapilas botón para PCB

- **Baterías tipo botón**

  Como baterías se eligen las pilas botón alcalinas de óxido de plata SR44 del fabricante RS. Tienen un diámetro de 11.6 mm y una altura de 5.4 mm. Su capacidad es de 165 mAh y su tensión nominal es de 1.55 V. Figura 3.21.

  Por un lado, y según lo explicado al seleccionar el regulador de tensión, el voltaje unificado de funcionamiento que se pretende es de 3.3 V, por lo que se tiene que alcanzar una tensión mayor con las baterías. Con 3 pilas SR44 en serie, se obtiene una tensión nominal de 4.65 V según la ecuación (3.7), así que el número de baterías necesario es de 3.

$$1,55\,V \cdot 3 = 4,65\,V \tag{3.7}$$

  Por otro lado, tenemos que la capacidad de la batería es de 165 mAh. En el apartado del regulador de tensión se hacía también una estimación del consumo aproximado de la carga que supone el circuito final, teniendo en cuenta sus tres componentes más importantes. Ese consumo es de aproximadamente 50 mA, según la ecuación (3.3). Por tanto, podemos estimar la autonomía aproximada del dispositivo; ver ecuación (3.8). Como 3 horas es un tiempo aceptable de funcionamiento para esta versión de la interfaz, se da por válida la elección de estas baterías.

$$\frac{165\,\text{mAh}}{50\,\text{mA}} = 3,3\,\text{h} \simeq 3\,\text{horas} \tag{3.8}$$

Figura 3.21: Baterías tipo botón

Capítulo 4

Diseño Hardware

En este capítulo se va a describir el desarrollo del dispositivo físico desde las ideas funcionales y la selección de componentes ya explicadas. La parte material y física (hardware), formada por el acelerómetro, el microcontrolador, el módulo Bluetooth y resto de componentes, se van a integrar en una placa de circuito impreso (PCB), la cual será montada sobre un soporte mecánico, formando todo ello el conjunto llamado dispositivo. La interfaz se completa con el software programado en el microcontrolador y el software instalado en el ordenador (PC), por lo que es en el dispositivo donde se realiza el diseño hardware.

Al ser un circuito digital, se necesitan pocos componentes auxiliares, reduciéndose así el número total de componentes, por lo que el mayor trabajo de diseño va a consistir en una adecuada colocación de dichos componentes para facilitar el conexionado. Asimismo, al realizar el diseño teórico hay que comprobar que todas las especificaciones y funcionalidades que se pedían a cada componente se cumplen realmente, asegurando el correcto conexionado entre sí.

En primer lugar se expone el diseño teórico del circuito que se plasma representado en el esquemático. Después se describe cómo se ha llevado a la realidad ese esquemático en una placa experimental, donde se realizan todas las pruebas y ensayos procedentes para realizar las mejoras o modificaciones necesarias antes de desarrollar la placa definitiva. Por último, se explica el diseño de la placa definitiva, el layout de la PCB. Se llega a un dispositivo físico final funcional con todos los componentes definitivos integrados en una sola placa.

4.1. Diseño del circuito. Esquemático

El diseño del circuito se puede ver en el Apéndice C. Hay un primer esquemático en el que se representa el circuito final utilizado para la placa experimental, y hay un segundo esquemático en el que se representa el diseño final, que es utilizado en la placa definitiva del dispositivo. Los dos esquemáticos son muy parecidos, siendo funcionalmente iguales, con la diferencia de que en el diseño de la placa definitiva se ha cambiado el oscilador de cristal por uno de montaje superficial, se ha añadido el led de control de alimentación con su respectiva resistencia y se ha unido de forma visual la línea Vcc. También se ha actualizado el nombre de alguno de los componentes.

Por supuesto, en el proceso de desarrollo del diseño ha habido otros esquemáticos intermedios que se han ido mejorando y afinando hasta llegar al diseño definitivo. Sin embargo, no son relevantes de cara a mostrar alguna fase intermedia del desarrollo hardware.

Por todo ello se toma como referencia a la hora de seguir la explicación del proceso de diseño el esquemático final de la placa definitiva, que aunque se puede ver en el Apéndice C, se reproduce también en la figura 4.1.

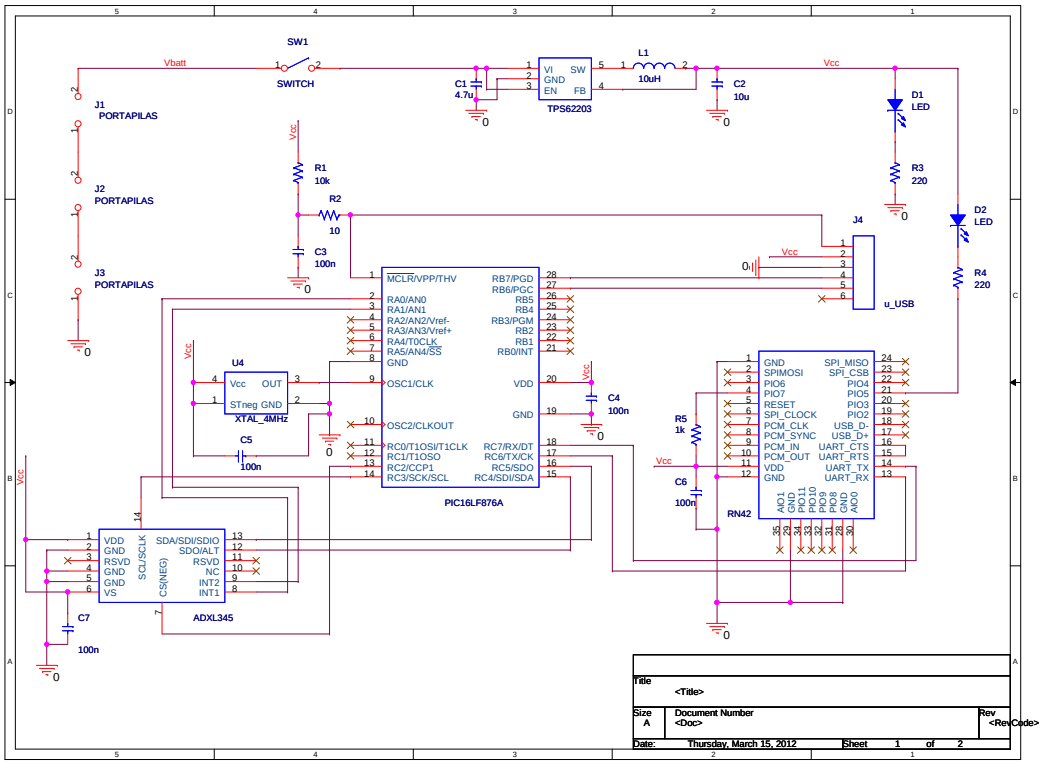


Figura 4.1: Esquemático definitivo

El diseño del circuito se realiza mediante el programa informático de diseño

electrónico OrCAD. El diseño del esquemático se realiza con la herramienta OrCAD Capture CIS. Este programa dispone de una gama muy amplia de representación de componentes electrónicos. Aun así, como los componentes principales seleccionados para el dispositivo son muy concretos, éstos no aparecen en la biblioteca de OrCAD, y se ha tenido que crear una representación personalizada de estos componentes desde cero, que se pueden ver en las figuras 4.2, 4.3, 4.4, 4.5 y 4.6.

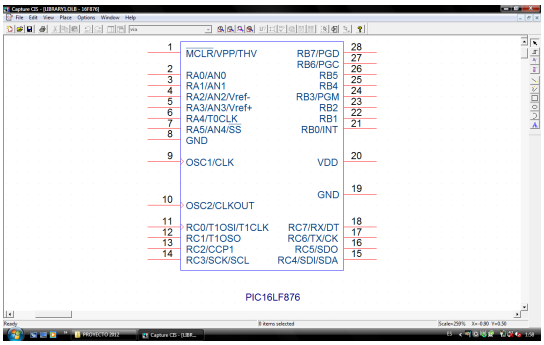


Figura 4.2: Componente para OrCAD del PIC16LF876A

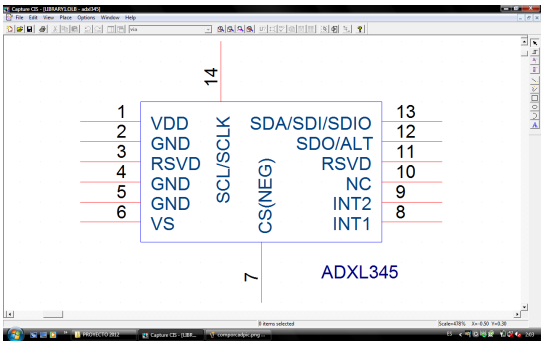


Figura 4.3: Componente para OrCAD del ADXL345

Se va a describir el diseño del circuito atendiendo a una separación por cinco bloques funcionales, que son los siguientes. Bloque de alimentación, bloque de LEDs de estado, bloque microcontrolador PIC16LF876A, bloque acelerómetro ADXL345 y bloque módulo Bluetooth RN42.

4.1.1. Bloque de alimentación

Las tres baterías, son representadas por tres portapilas, puesto que de cara a la representación esquemática en OrCAD hay dos orientaciones, el diseño para hacer una simulación del circuito en OrCAD o el diseño para transferir el esquemático posteriormente a un diseño layout para PCB. Como nuestra

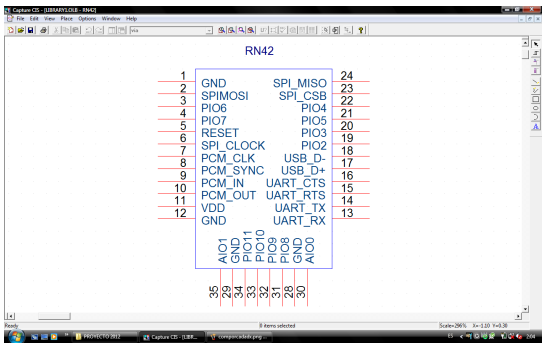


Figura 4.4: Componente para OrCAD del RN42

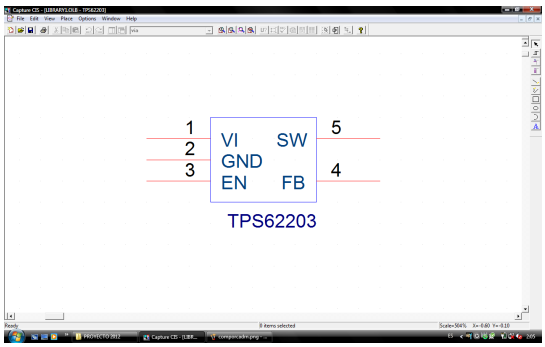


Figura 4.5: Componente para OrCAD del TPS62203

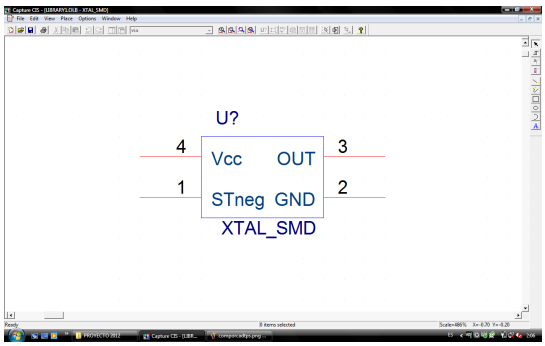


Figura 4.6: Componente para OrCAD del XTAL

orientación es la segunda, representamos las baterías con sus consiguientes receptáculos, que de cara al diseño layout, representan los contactos a masa (0) y a Vbatt. La línea que sale de los portapilas es Vbatt, que es la tensión que hay desde un extremo de las tres pilas en serie hasta masa. En esta línea se sitúa el interruptor (switch), que abre y cierra el circuito según control externo.

A continuación, el regulador de tensión TPS62203 regula el voltaje de entrada Vbatt, dejándolo en el valor ya explicado anteriormente de 3.3 V. Este es el valor de tensión de alimentación, que llamamos Vcc. Para que el regulador funcione correctamente necesita tres pasivos que son descritos en la hoja de características (ver Apéndice E). Dos condensadores en paralelo, uno de 4,7 μF a la entrada y otro de 10 μF a la salida. Además necesita una bobina o inductancia de 10 μH en serie en la salida, pin 5 (SW). El resto de conexiones del componente se puede ver tanto en el esquemático como en la hoja de características, y una descripción de los pines en la tabla 4.1. En la figura 4.7 se extrae el bloque de alimentación del conjunto del esquemático.

Tabla 4.1: Resumen pines TPS62203

TPS62203

Nº de pin	Nombre del pin	Descripción
1	VI	Voltaje de entrada (Vin)
2	GND	Masa
3	EN	Habilita llevado a Vin
4	FB	Conectar a salida si Vo es fijo
5	SW	Conectar a inductancia

4.1.2. Bloque de LEDs de estado

Partiendo de la línea de alimentación Vcc se habilitan dos diodos LED. Un LED de alimentación (D1) de color rojo, para controlar el estado de encendido o apagado del dispositivo. Este LED lleva su correspondiente resistencia de 220 Ω y es llevado a masa. Por otro lado, el LED de control de estado del RN42 (D2) sirve para hacer un seguimiento del estado de conexión del módulo Bluetooth. Lleva su correspondiente resistencia de 220 Ω y va conectado al pin 21 (PIO5) del RN42. El funcionamiento del estado de este LED se explicará más adelante. La elección del valor de las resistencias se hace con el cálculo de las ecuaciones (4.1) y (4.2), que salen de la figura 4.8.

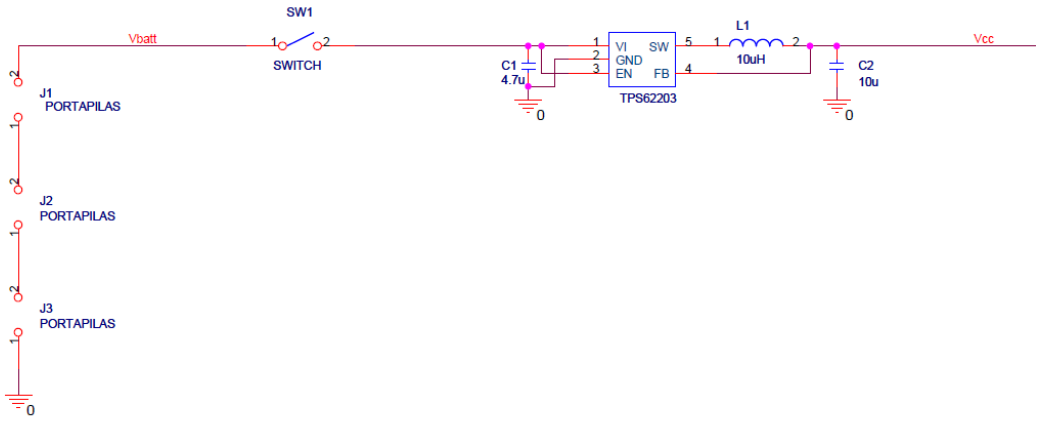


Figura 4.7: Bloque de alimentación

Diodo rojo D1 : Vf=2.2 V ; If=50 mA (hoja de características)

$$R = \frac{V_{cc} - V_f}{I_f} \tag{4.1}$$

$$R = \frac{3,3 - 2,2\,\mathrm{V}}{50\,\mathrm{mA}} = \frac{1,1}{50} = 0,022\,k\Omega = 22\,\Omega \tag{4.2}$$

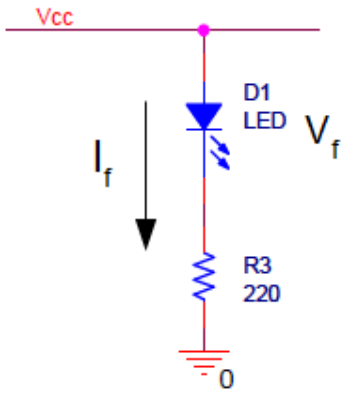


Figura 4.8: Cálculo de la resistencia del LED

Sin embargo, elegir esta resistencia significa que su consumo sería de 50 mA. Teniendo en cuenta como ya hemos visto que el conjunto de los componentes principales consumen aproximadamente 50 mA, esto haría que una sola resistencia consumiera lo mismo que todo el resto del circuito. Como no importa que el LED brille menos, se reduce su corriente de paso en un factor de 10, de forma que consuma 5 mA, un valor aceptable para el conjunto. Esto hace que la resistencia asociada aumente su valor también en un factor de 10, quedando

una resistencia de 220 Ω; ecuación (4.3). Además, el LED elegido de montaje superficial es de alto brillo, por lo que la afección en su rendimiento lumínico no será apreciable.

$$\frac{50\,\text{mA}}{10} = 5\,\text{mA} \Rightarrow \frac{3,3 - 2,2\,\text{V}}{5\,\text{mA}} = 220\,\Omega \tag{4.3}$$

Hay que señalar que para el LED verde (D2) según la hoja de características, Vf=3,9 V con una If=30 mA. Como en este caso $V_f > V_{cc}$, no se puede calcular la resistencia directamente. Mirando la gráfica Vf-If de la hoja de características para el LED verde se puede ver que para una V=3,3 V, la corriente If=10 mA, lo que sería simulando una resistencia de 0 Ω. Esto nos indica el consumo de la resistencia que se ponga. Así, colocamos una resistencia de 220 Ω como en el otro LED para tener un orden de magnitud similar en los valores. El resultado sería tener menos brillo que en el otro LED puesto que $220\,\Omega > 0\,\Omega$ y $10\,\text{mA} > 5\,\text{mA}$, sin embargo de nuevo al ser un LED de alto brillo no se va a notar la diferencia. En la figura 4.9 se ve el bloque de LEDs de estado.

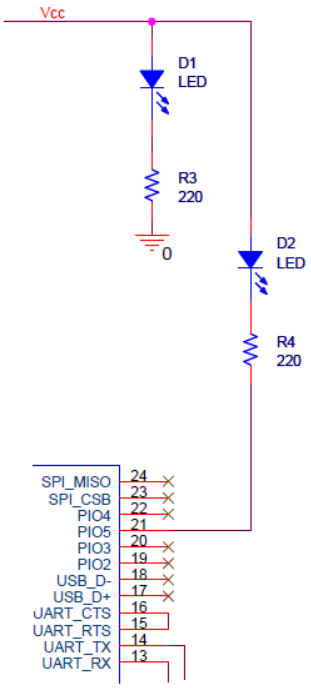


Figura 4.9: Bloque de LEDs de estado

4.1.3. Bloque Microcontrolador PIC16LF876A

La alimentación del PIC va directa a Vcc=3.3 V. Notar que en OrCAD, todas las líneas del esquemático que tengan el mismo nombre están unidas. Es decir, todas las líneas llamadas Vcc están conectadas entre sí, es un solo punto eléctrico con la misma tensión, aunque en el esquemático no se muestre. Para estabilizar esta tensión, se añade un condensador de desacoplo de 100 nF en paralelo entre Vcc (pin 20 (VDD) del PIC) y masa ó 0 (pin 19 (GND) del PIC), y se coloca lo más cerca posible del microcontrolador.

Para poder funcionar, el PIC necesita una fuente de reloj externa. En el pin 9 (OSC1/CLK) se conecta en la versión definitiva un oscilador de cristal de 4 MHz, que funciona a modo de reloj externo puesto que el componente definitivo de montaje superficial tiene un solo pin de salida (ver hojas de características del PIC y del cristal). En la placa experimental este componente es un resonador de cristal al uso y se conecta en paralelo a los pines 9 y 10 del PIC. De los cuatro pines del oscilador definitivo, los pines 1 y 4 van a Vcc y el pin 2 a masa. Entre ambos puntos eléctricos se coloca un condensador de 100 nF lo más cerca posible del cristal para estabilizar la señal. El pin 3 es la salida que entrega la señal de reloj al PIC.

Se añade un circuito de programación asociado a los pines 1, 27 y 28 del PIC para poder utilizar la función In-Circuit Serial Programming (ICSP) de Microchip. Esta herramienta permite programar el PIC incluso estando instalado en el circuito, precisamente a través de estos pines. La programación se realiza mediante el programador MPLAB ICD2 via USB, el cual va conectado a su vez a un ordenador con el programa MPLAB IDE que controla todo el proceso. El pin 1 (MCLR/VPP/THV) se conecta al pin 1 del conector microUSB. Cuando el programador se conecta, esta línea da la tensión de programación Vpp, llevando al PIC al modo programación. El pin 27 (RB6/PGC) del PIC va conectado al pin 5 del conector, por el cual va la señal de reloj de programación. Por el pin 28 (RB7/PGD) va la señal de datos de programación, el cual está conectado al pin 4 del conector. A su vez los pines 2 y 3 del conector deben ir conectados a Vcc y masa respectivamente. Según el manual de usuario del MPLAB ICD2 [39], el pin 1 debe llevar una resistencia de pull-up de 10 k para no limitar la capacidad de reset, y a la vez no puede estar conectado directamente a un condensador porque necesita transiciones rápidas de 0 a Vpp cuando comienza una programación, por tanto, añadimos la resistencia de 10 Ω al condensador de 100 nF de la forma descrita en el esquemático. Ver figuras 4.10, 4.11 y 4.12.

El PIC va conectado al acelerómetro ADXL345 a través de una conexión serie síncrona SPI y también se conectan entre sí 2 pines para detectar las interrupciones. El pin 2 (RA0/AN0) y el pin 3 (RA1/AN1) del PIC se conectan a los pines 8 (INT1) y 9 (INT2) del ADXL345 respectivamente, para tener la

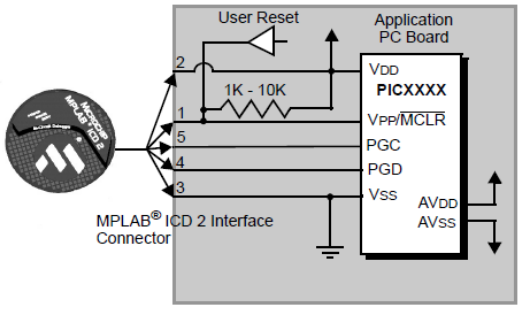


Figura 4.10: Conexiones del programador MPLAB ICD2 con el PIC

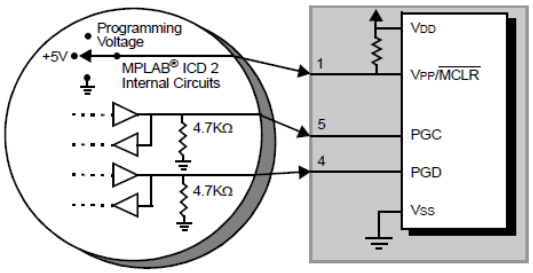


Figura 4.11: Conexiones internas del MPLAB ICD2

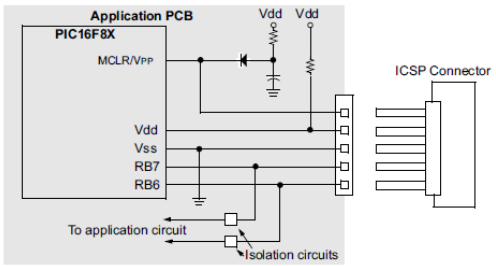


Figura 4.12: Conexiones generales para ICSP con MPLAB ICD2

posibilidad de detectar 2 interrupciones si así se considerase necesario. Por otro lado, tanto el PIC como el ADXL345 tienen posibilidad de conectarse en serie mediante los modos SPI e I^2C. Se elige como conexión serie el modo SPI, en lugar del modo I^2C, principalmente porque el conexionado es más sencillo y necesita menos componentes extra, lo que ahorra espacio, consumo y peso. A su vez, se elige el modo de conexión a 4 cables. Por tanto, la señal de habilitación CSneg conecta el pin 13 (RC2/CCP1) del PIC con el pin 7 (CSneg) del ADXL345. La señal de reloj Serial-Clock conecta el pin 14 (RC3/SCK/SCL) del PIC con el pin 14 (SCL/SCLK) del acelerómetro. La señal de datos entrante al microcontrolador Serial-Data-In conecta el pin 15 (RC4/SDI/SDA) del PIC al pin 12 (SDO/ALT) del acelerómetro. La señal de datos saliente del microcontrolador Serial-Data-Out conecta el pin 16 (RC5/SDO) del PIC al pin 13 (SDA/SDI/SDIO) del ADXL345. Este esquema de conexiones se puede ver en la figura 4.13. Notar que el pin 13 del PIC es un pin genérico de entrada/salida que se establece como salida para generar la señal CSneg por software, sin embargo los pines 14,15 y 16 del PIC sí que se utilizan en su función especial de pines del puerto SPI (SCK, SDI y SDO).

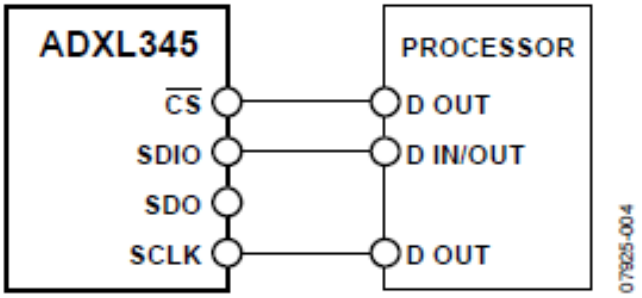


Figure 34. 3-Wire SPI Connection Diagram

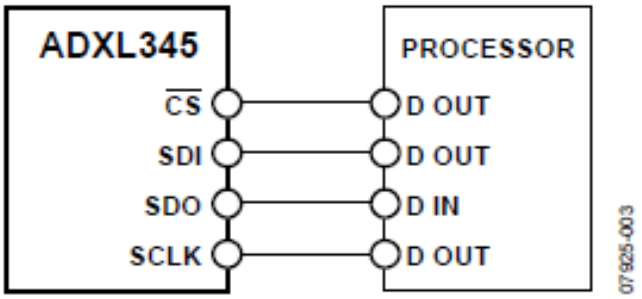


Figure 35. 4-Wire SPI Connection Diagram

Figura 4.13: Conexiones SPI del ADXL345

La conexión con el módulo Bluetooth RN42 se realiza desde el puerto serie

asíncrono USART del PIC hasta el puerto USART del RN42. La señal saliente Transmit (TX) conecta el pin 17 (RC6/TX/CK) del PIC al pin 13 (UART_RX) del RN42. A su vez la posible señal entrante Receive (RX) conecta el pin 18 (RC7/RX/DT) del PIC con el pin 14 (UART_TX) del RN42. Según la hoja de características del RN42 si la alimentación de éste y del PIC son distintas hay que poner las resistencias adecuadas formando un circuito divisor de tensión para adecuar la diferencia de tensión de ambos componentes, y que la señal enviada a través de las líneas TX/RX no se vea afectada. Esto se debe a que con distintas tensiones en microcontrolador y módulo Bluetooth, el '1' lógico sería de distintos voltajes y habría incompatibilidad. Precisamente, al haber unificado la alimentación de todos los componentes a un mismo valor (3.3 V), esto hace que no sea necesario añadir ningún pasivo y se puedan conectar directamente las líneas TX/RX. Es importante también que ambos componentes tengan la misma masa, el mismo '0' lógico, cuestión que se cumple puesto que todos los componentes del circuito están integrados en la misma placa con la misma masa.

Para terminar con el PIC, falta por citar el pin 8 (GND), que es redundante al pin 19 y también se conecta a masa. Los demás pines que no se han descrito quedan abiertos, sin conexión, puesto que no se van a utilizar. Se comprueba que como son todos pines entrada/salida de los puertos A,B ó C, no hay problema por dejarlos 'al aire'. La excepción es el pin 10, que no es genérico, pero se comprueba en la hoja de características que al conectar el oscilador de cristal como fuente de reloj externa, se puede dejar este pin sin conexión sin causar conflicto. En la tabla 4.2 están resumidas las funciones de cada pin del PIC. La figura 4.14 muestra el esquemático del bloque del microcontrolador.

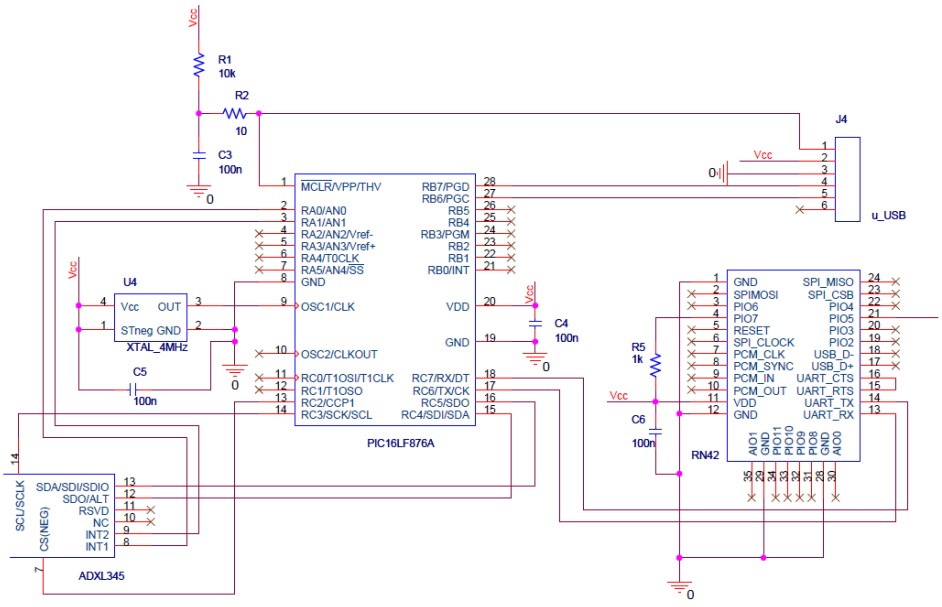


Figura 4.14: Bloque Microcontrolador PIC16LF876A

Tabla 4.2: Resumen pines PIC16LF876A

PIC16LF876A

Nº de pin	Nombre del pin	Descripción
1	MCLR/VPP/THV	Pin voltaje de programación
2	RA0/AN0	I/O digital – Entrada
3	RA1/AN1	I/O digital – Entrada
8	GND	Masa
9	OSC1/CLK	Entrada de reloj externo
13	RC2/CCP1	I/O digital – Salida
14	RC3/SCK/SCL	Salida Serial Clock SPI
15	RC4/SDI/SDA	Entrada Serial Data In SPI
16	RC5/SDO	Salida Serial Data Out SPI
17	RC6/TX/CK	Salida Transmit USART
18	RC7/RX/DT	Entrada Receive USART
19	GND	Masa
20	VDD	Tensión de alimentación
27	RB6/PGC	Señal de reloj de programación
28	RB7/PGD	Señal de datos de programación

4.1.4. Bloque Acelerómetro ADXL345

Las conexiones principales del ADXL345 ya han sido explicadas en el bloque del microcontrolador, siendo éstas la conexión SPI y las interrupciones. Desde el punto de vista del acelerómetro, y ya que según la aplicación, éste es el que toma los datos físicos y los transmite al PIC, las señales SPI son CSneg entrante (pin 7), SCLK entrante (pin 14), SDO saliente (pin 12) y SDI entrante (pin 13). Este último no se utilizará puesto que todo el flujo de datos es saliente desde el acelerómetro al microcontrolador, pero se deja conectado. En cuanto a las interrupciones se conectan el pin 8 y 9 (INT1 e INT2), salientes, con el PIC para enviar la señal de interrupción que se establezca por software si ésta se produce. Por otro lado, el pin 1 (VDD) y el pin 6 (VS) se conectan a Vcc, mientras que los pines 2, 4 y 5 (GND) se conectan a masa. Entre ambos puntos eléctricos se añade un condensador en paralelo de 100 nF lo más cercano posible al acelerómetro para estabilizar y desacoplar de ruidos la señal de alimentación Vcc. Los pines 3 (RSVD), 10 (NC) y 11 (RSVD) se dejan sin conectar. El esquemático en torno al ADXL345 se ve en la figura 4.15 y se muestra el resumen de sus pines en la tabla 4.3.

Tabla 4.3: Resumen pines ADXL345

ADXL345

Nº de pin	Nombre del pin	Descripción
1	VDD	Tensión de alimentación digital
2	GND	Masa
4	GND	Masa
5	GND	Masa
6	VS	Tensión de alimentación
7	CS(NEG)	Chip Select
8	INT1	Salida interrupción 1
9	INT2	Salida interrupción 2
12	SDO/ALT	Salida Serial Data SPI
13	SDA/SDI/SDIO	Entrada Serial Data SPI
14	SCL/SCLK	Serial Clock SPI

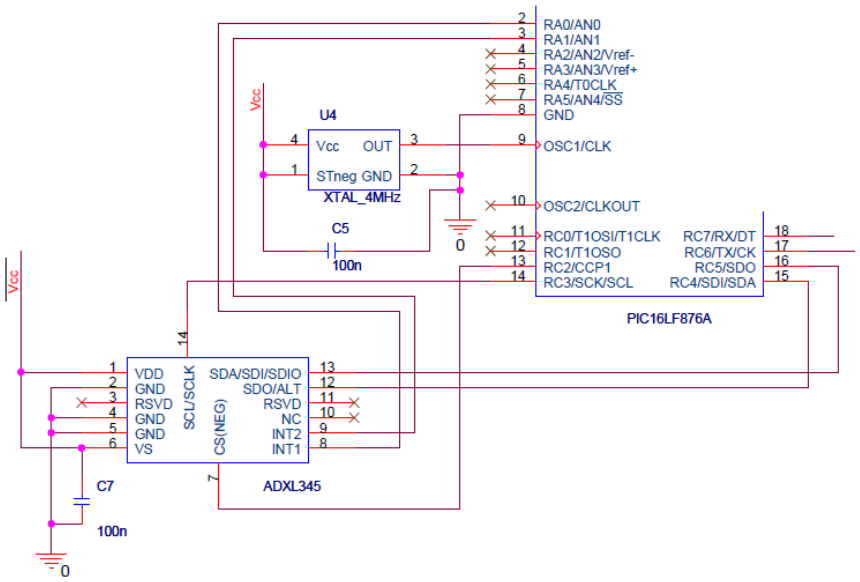


Figura 4.15: Bloque Acelerómetro ADXL345

4.1.5. Bloque Módulo Bluetooth RN42

La conexión principal del RN42 ya ha sido explicada en el bloque del microcontrolador, que es la recepción vía serie de datos por el pin 13 (UART_RX). El pin 14 (UART_TX) para transmitir datos no se va a utilizar ya que en esta aplicación el sentido de los datos es exclusivamente saliente del microcontrolador al módulo Bluetooth, aún así lo dejamos correctamente conectado con el PIC. Para terminar con el puerto USART del RN42, conectamos entre sí cortocircuitando los pines 15 (UART_RTS) y 16 (UART_CTS), ya que la conexión serie PIC–RN42 ha sido de 3 cables (tx/rx/gnd); ver hoja de características. En cuanto a la alimentación, el pin 11 (VDD) es conectado a Vcc y los pines 1,12,28 y 29 (GND) son conectados a masa. Como siempre, conectamos un condensador estabilizador de desacoplo en paralelo de 100 nF. El pin 4 (PIO7) lleva una resistencia de pull-up de 1k hasta Vcc. Con esto, llevamos el pin a nivel alto, y es la forma de configurar por hardware la tasa o velocidad de transmisión de datos del RN42, estableciendo el valor en 9600 baudios. Este valor es suficiente para conectarse vía bluetooth con el ordenador, el valor por defecto si no se coloca la resistencia es de 115 kbps, que es demasiado rápido para esta aplicación. El pin 21 (PIO5) lleva conectado el LED verde con la resistencia de 220 Ω que ya se ha explicado en el bloque de LEDs de estado. Este LED tiene varios modos de funcionamiento según el estado de conexión en el que se encuentre el módulo Bluetooth, que se puede ver en la tabla 4.4. Parpadea a distintas velocidades según el modo sea de configuración, de inicio, o de modo discovery para buscar emparejamiento. Una vez el módulo ha encontrado o ha sido encontrado por otro dispositivo Bluetooth al cual se conecta y empareja,

el LED pasa de parpadear a tener luz verde fija.

Tabla 4.4: Modos del LED verde asociado al estado de conexión del RN42

MODE	GREEN LED blink rate
Configuring	10 times per second
Startup/Config Timer	2 times per second
Discoverable/Inquiring/Idle	Once per second
Connected	Solid ON

El resto de pines que no se han nombrado se dejan sin conectar, comprobándose que no se crea conflicto por ello. La tabla 4.5 muestra la función de los pines del RN42. La figura 4.16 muestra el bloque del módulo Bluetooth.

Tabla 4.5: Resumen pines RN42

RN42

Nº de pin	Nombre del pin	Descripción
1	GND	Masa
4	PIO7	Baud rate (high=9600 baudios)
11	VDD	Tensión de alimentación
12	GND	Masa
13	UART_RX	Entrada Receive UART
14	UART_TX	Salida Transmit UART
15	UART_RTS	Deshabilita transmisión a nivel alto
16	UART_CTS	Deshabilita transmisión a nivel alto
21	PIO5	Estado de conexión
28	GND	Masa
29	GND	Masa

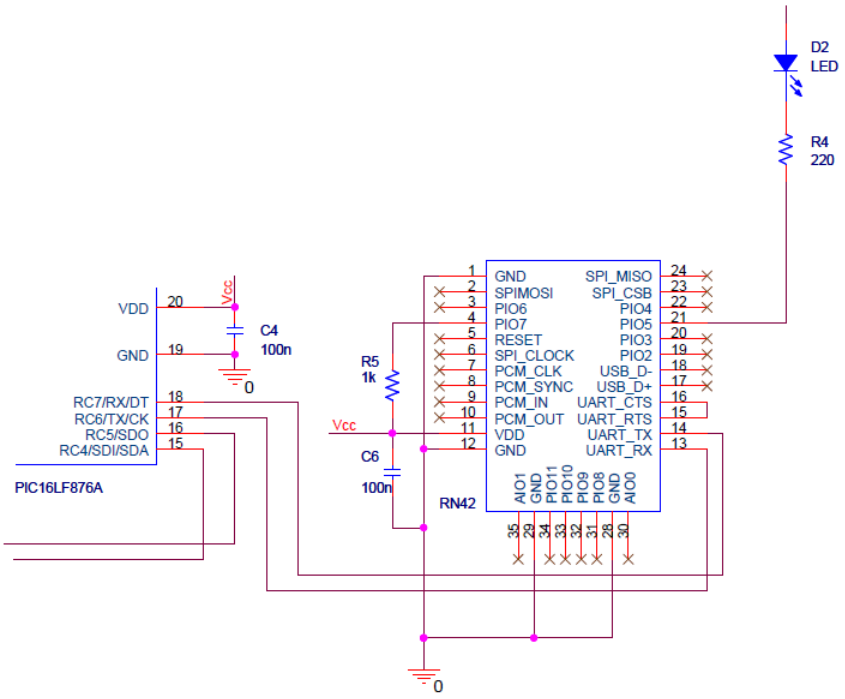


Figura 4.16: Bloque Módulo Bluetooth RN42

Con esto se ha descrito con detalle el circuito en su totalidad, el esquemático. Hay que reseñar que puesto que el microcontrolador es el cerebro del circuito, y alrededor de él giran todos los demás componentes, a la hora del diseño del esquemático, el PIC se ha colocado en el centro para facilitar las conexiones. Lo siguiente a tener en cuenta es que el oscilador de cristal tiene que ir lo más cercano posible al PIC ya pensando en el montaje físico. Además, como los pines de conexión serie SPI del PIC están abajo y los pines de programación están arriba, el ADXL345 se coloca debajo y el circuito de programación encima. Los pines del puerto USART del PIC están en la parte derecha, por ello, se coloca el RN42 a la derecha. Por último el bloque de alimentación se coloca arriba del todo para englobar al resto de componentes y facilitar el conexionado.

4.2. Placa experimental

Antes de fabricar la placa definitiva, que por sus pequeñas dimensiones y complejidad va a resultar un proceso largo y costoso, se construye una placa experimental para poder hacer todas las pruebas necesarias en laboratorio, comprobar la funcionalidad del diseño teórico y realizar todas las modificaciones o

mejoras que sean oportunas. Además, todo el proceso de diseño del software desde cero necesita de multitud de pruebas que se van a acometer mejor en una placa de pruebas barata y rápida. En la figura 4.17 se puede ver el aspecto final de esta placa.

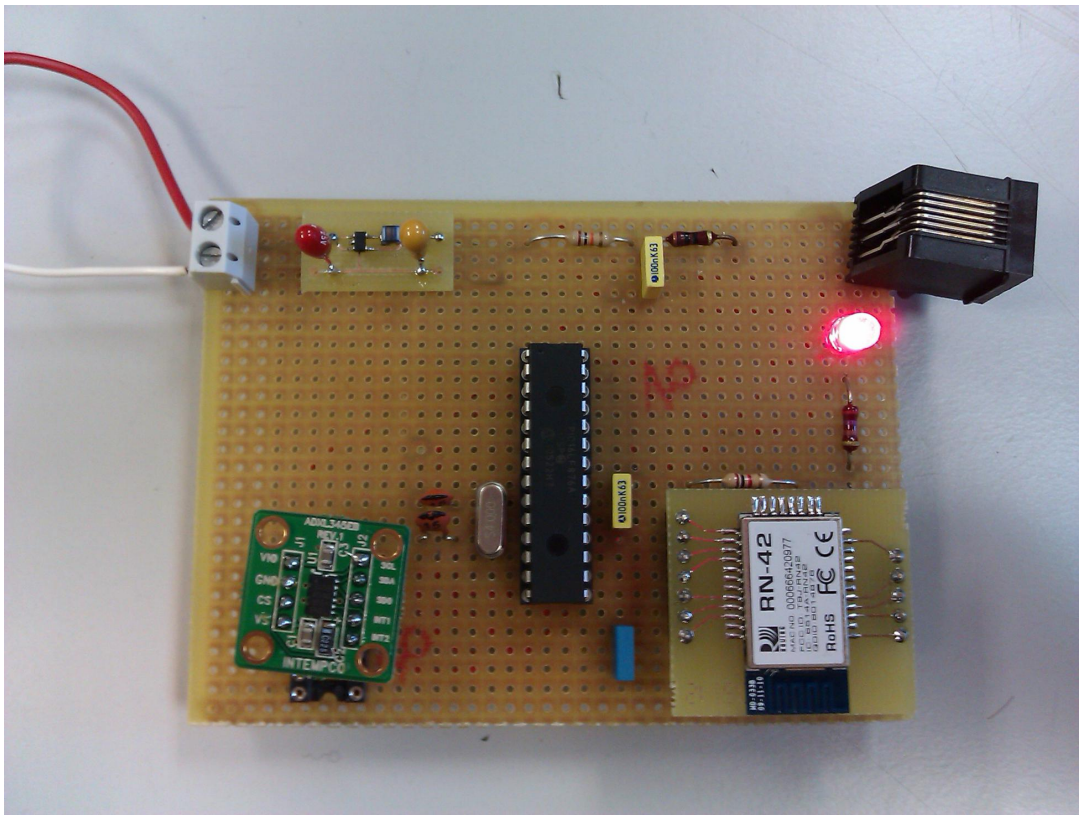


Figura 4.17: Placa experimental

En primer lugar hay que tener en cuenta que en esta placa no importan las dimensiones, lo importante es mantener la funcionalidad, es decir, los componentes tienen que ser los mismos que en la placa definitiva en cuanto a su función aunque sean más grandes, y las conexiones se deben de mantener igual, siguiendo el esquemático. Por tanto a la hora del montaje se sigue el esquemático de la placa experimental del Apéndice C, ya se han explicado las diferencias con el esquemático definitivo. El listado de componentes completo de esta placa experimental se puede ver en el Apéndice B.

El montaje se realiza en una placa de puntos, que se recorta a la medida deseada, calculando el sitio que ocupan todos los componentes y dejando holgura para no estar apretado. Todos los componentes son de montaje pasante y se sueldan en la cara posterior de la placa de puntos, aunque hay varias particularidades y excepciones.

El acelerómetro ADXL345 es de montaje superficial y demasiado pequeño

para ser manejado en una placa de puntos, así que se compra la placa de evaluación propia de Analog Devices para este componente, que es una placa de circuito impreso con todos los pines accesibles y pasantes. Esta placa se inserta en 2 zócalos superpuestos puesto que las dos filas de pines accesibles de la placa no están alineadas y con un solo zócalo no encajaba. El resultado es que los ejes del acelerómetro no están alineados con los ejes de la placa de puntos, hay una ligera desviación, aunque esto se tiene en cuenta más adelante a la hora de realizar las pruebas de forma correcta; figura 4.18. La placa de evaluación incluye el condensador de desacoplo de 100 nF.

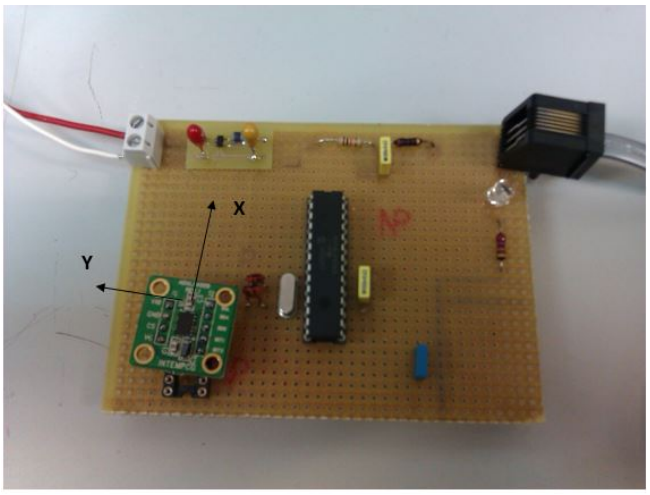


Figura 4.18: Ejes desalineados del acelerómetro respecto de la placa experimental

El módulo Bluetooth RN42 también es de montaje superficial, no tiene versión de montaje through-hole o pasante y tampoco tiene placa de evaluación de la marca, así que se crea un layout de PCB personalizado para este componente. Esta placa propia luego es montada sobre un zócalo que se suelda a la placa de puntos. Para diseñar este layout se utiliza el programa OrCAD Layout Plus. Antes, con el programa OrCAD Capture CIS, se diseña el circuito eléctrico que componen el RN42 y sus correspondientes puntos eléctricos para los agujeros pasantes; figura 4.19. Una vez transferido este diseño a OrCAD Layout Plus, aquí se tiene que diseñar una footprint o huella del componente precisa a partir de las especificaciones de dimensiones de layout de la hoja de características; figura 4.20. Es imprescindible ser totalmente riguroso con las dimensiones de los pads o isletas en el diseño de la huella. Con este footprint del componente y el de los agujeros pasantes, junto con el diseño eléctrico transferido anteriormente, se procede al rutado de las pistas que conectan los pines que queremos usar del RN42 con sus correspondientes pads pasantes que alineamos entre sí y en dos filas paralelas con el espaciado exacto de separación de los puntos de la placa de puntos. El resultado final se puede ver en las figuras 4.21, 4.22, 4.23 y 4.24. La fabricación de esta PCB y el montaje con el RN42 soldado se realiza en el laboratorio técnico del Departamento de Sistemas y Automática de la Universidad,

gracias a la labor de Fernando San Deogracias y Ángela Nombela.

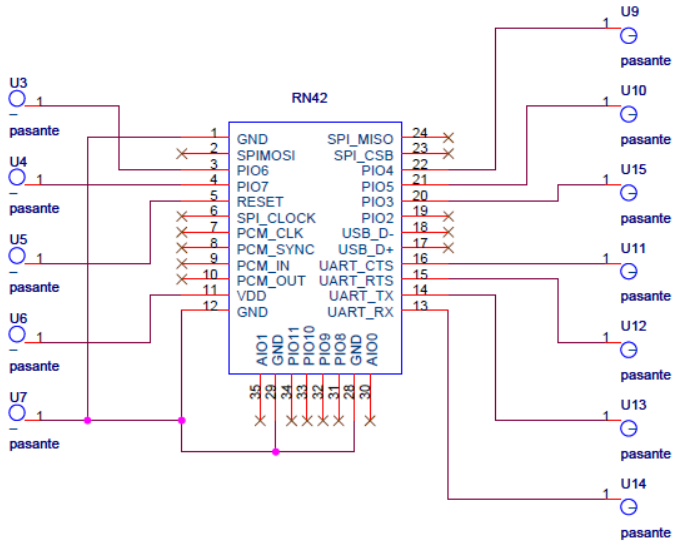


Figura 4.19: Diseño en OrCAD Capture CIS del esquema eléctrico para la PCB del RN42

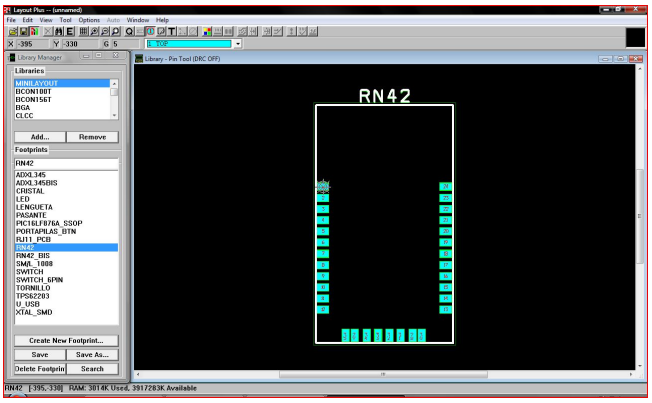


Figura 4.20: Footprint del RN42

El regulador de tensión TPS62203 sigue el mismo proceso que el módulo Bluetooth, ya que solo hay versión del componente con montaje superficial. Al diseño del layout de la PCB personalizada para este componente hay que añadir la inductancia de 10 μH que también es de montaje superficial, y los dos condensadores de 4.7 y 10 μF, para crear una placa integrada del regulador con sus pasivos asociados. Mismo proceso: se diseña en OrCAD Capture CIS un esquemático con los componentes y los puntos eléctricos que servirán de agujeros pasantes para soldar a la placa de puntos; figura 4.25. Después se crea en OrCAD Layout Plus la huella o footprint del TPS62203 con las dimensiones exactas de los pads sacado de la hoja de características; figura 4.26. A continuación se transfiere el circuito de Capture CIS a Layout Plus, y se procede al rutado

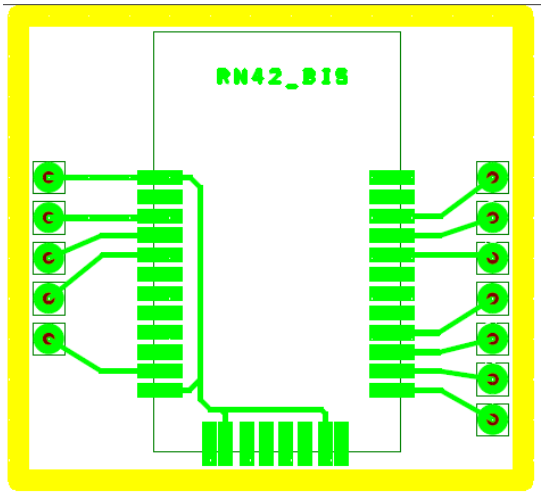


Figura 4.21: Layout PCB para el RN42

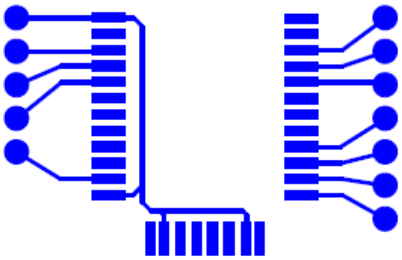


Figura 4.22: Fotolito capa TOP de PCB para RN42

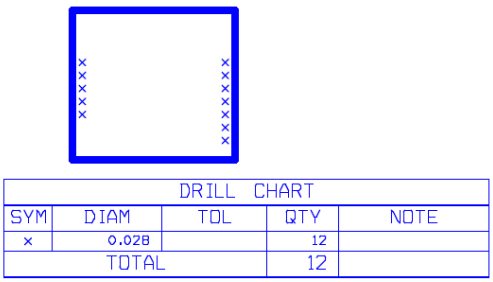


Figura 4.23: Fotolito Drill Chart de PCB para RN42

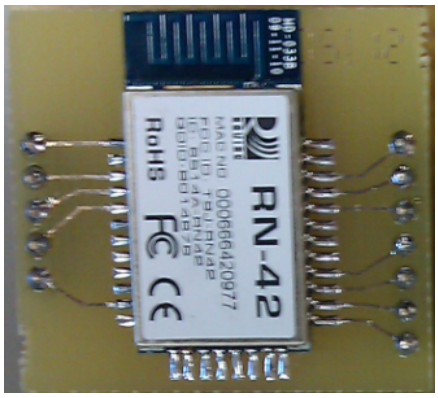


Figura 4.24: Foto de la PCB con el RN42 montado

del conjunto, resultando el layout de la figura 4.27. Esta PCB se fabrica en la Universidad también y el montaje de regulador, bobina y condensadores es realizado de nuevo por Fernando de manera excelente. Se puede ver su aspecto final en la figura 4.28.

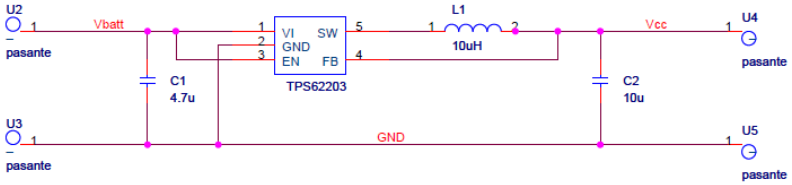


Figura 4.25: Diseño en OrCAD Capture CIS del esquema eléctrico para la PCB del TPS62203

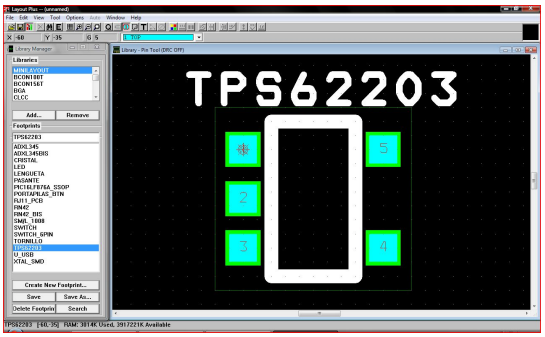


Figura 4.26: Footprint del TPS62203

El microcontrolador PIC16LF876A de montaje through-hole o pasante es montado sobre un zócalo que va soldado a la placa de puntos. A su lado se coloca el oscilador de cristal pasante de 4 MHz, que en este caso va acompañado de dos condensadores de 15 pF en paralelo (ver esquemático de la placa experimental). Al lado del PIC también se coloca su condensador estabilizador de 100 nF. El circuito programador formado por las resistencias de 10 kΩ y 10 Ω,

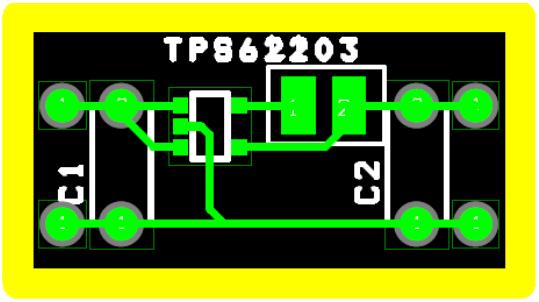


Figura 4.27: Layout PCB para el TPS62203

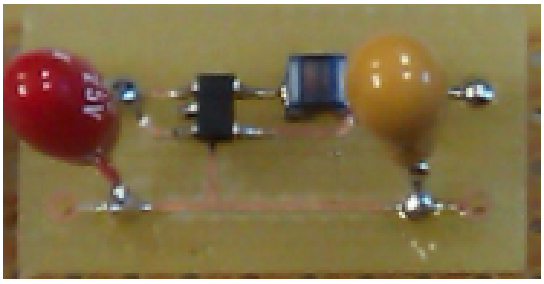


Figura 4.28: Foto de la PCB con el TPS62203 y sus pasivos asociados montados

el condensador de 100 nF y el conector para programar el PIC, queda situado arriba a la derecha. Como estamos en fase experimental, el conector es un RJ11 hembra pasante, que es el tipo de conexión que utiliza el programador MPLAB ICD2.

Alrededor del RN42 situamos el resto de los componentes que faltan. El diodo LED pasante con su respectiva resistencia de 220 Ω es el D2 del esquemático, y controla el estado de conexión del módulo Bluetooth. Aunque aquí es de color rojo, después en el dispositivo definitivo este LED será de color verde, y es el LED al que en apartados anteriores hemos denominado LED verde de control de estado del RN42. También colocamos la resistencia de 1 kΩ asociada al pin PIO7 del RN42 cerca de éste, y el condensador (azul) de 100 nF que estabiliza la alimentación en este módulo.

Por último, señalar que la alimentación de la placa experimental no se realiza con baterías, sino directamente desde la fuente de alimentación del laboratorio del Departamento de Sistemas y Automática de la Universidad, ajustada a un voltaje de 4.5 V. Se suelda una clema de 2 vías como entrada de alimentación de la placa, a la que va conectado el polo positivo (+) y la masa (- ó 0) de la fuente de alimentación. El interruptor o switch del esquemático, es directamente en este caso el interruptor de encendido de la fuente de alimentación. Y finalmente decir que el LED rojo de control de estado de la alimentación (con su resistencia de 220 Ω) no se monta en esta placa experimental, puesto que esta función la hace el piloto de encendido también de la fuente de alimentación del laboratorio.

Figura 4.29.

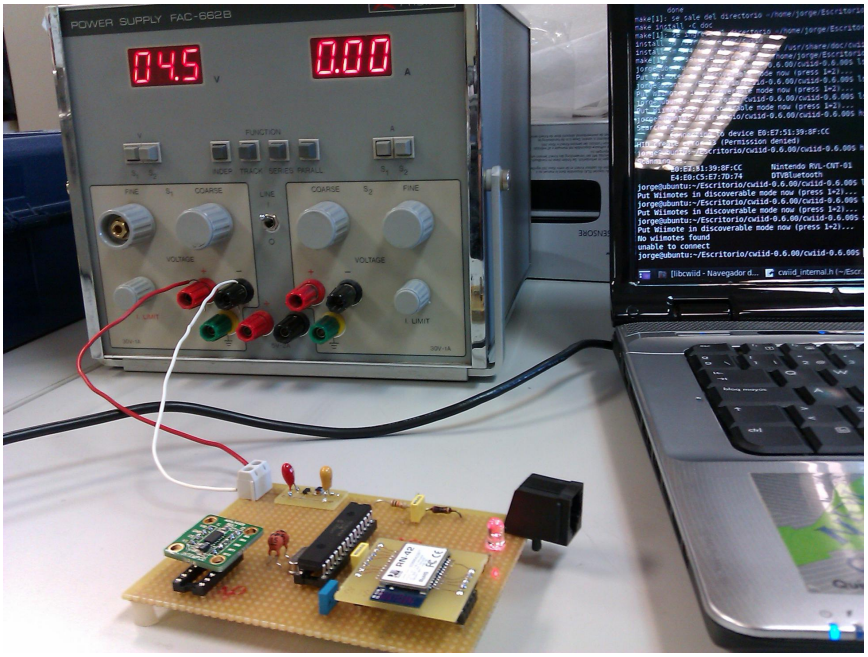


Figura 4.29: Placa experimental con la fuente de alimentación del laboratorio

El trabajo más delicado una vez realizado el montaje de los componentes en la placa, es el conexionado eléctrico en la cara posterior para completar todas las conexiones del circuito representados en el esquemático. Antes de soldar definitivamente los componentes a la placa, se ha estudiado minuciosamente la colocación de éstos para asegurar que se van a poder realizar todas las conexiones sin conflicto en la cara posterior. Con soldador, estaño, cables y mucho trabajo, se consiguen conectar todos los pines que tienen función en los componentes principales con todo el resto de componentes pasivos y auxiliares del circuito; ver figura 4.30. Se realiza un minucioso trabajo de comprobación de la continuidad con el polímetro para asegurar que todas las conexiones son las correctas y no hay cortocircuitos.

4.2.1. Fase 2 de la placa experimental

Con la placa experimental explicada hasta ahora se realizan todas las pruebas de hardware y software que se explican en el capítulo 7 del documento. Sin embargo, una vez comprobado que se llega a obtener el funcionamiento deseado, y antes de fabricar la placa definitiva, es necesario comprobar que funcionan correctamente los componentes de la placa definitiva que no han sido montados en la placa experimental, que son el oscilador de cristal de 4 pines y el conector microUSB hembra, figura 4.31.

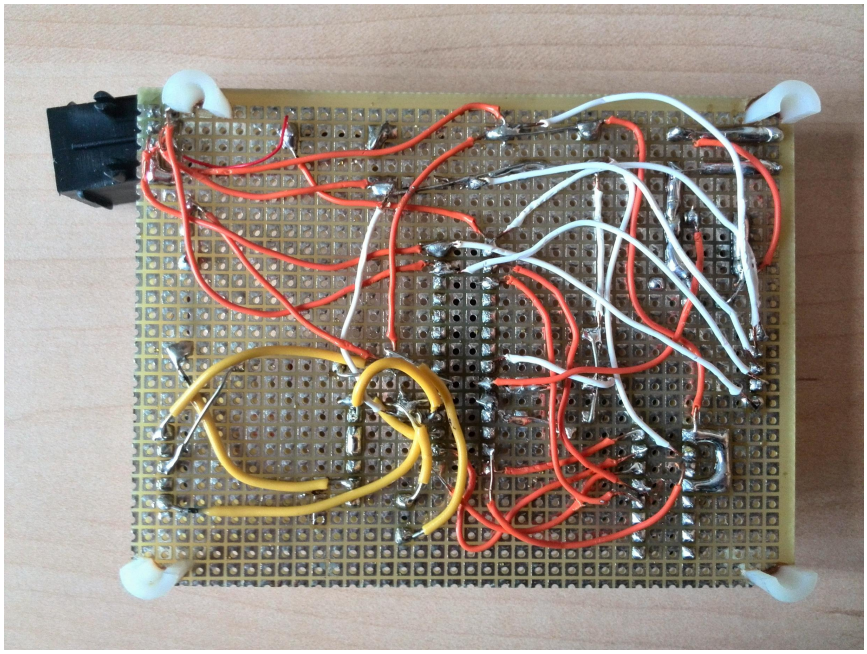


Figura 4.30: Parte posterior de la placa experimental

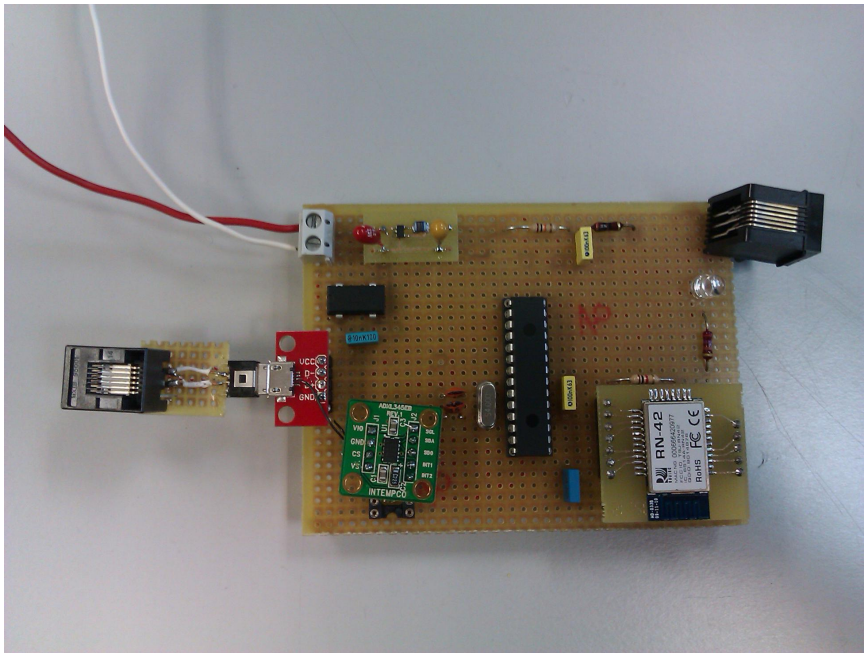


Figura 4.31: Fase 2 de la placa experimental

En cuanto al oscilador de cristal de 4 MHz, hasta ahora se había montado un resonador de cristal de 2 pines genérico, conectado al PIC a través de los pines 9 y 10. Este resonador se desconecta del circuito junto con sus 2 condensadores asociados de 15 pF, y se monta en la placa el oscilador de cristal definitivo EPSON SG8002DC, aunque en su versión de montaje pasante para soldarlo en la placa de puntos. Asociado a él se monta un condensador de 100 nF (ver esquemático de la placa definitiva en el Apéndice C). Es importante probar que funciona correctamente puesto que ahora se conecta al PIC sólo a través del pin 9, funcionando como fuente de reloj externa, y esta característica no se había probado.

El conector hembra microUSB se monta en la placa de puntos a través de una mini-placa PCB del fabricante Sparkfun para tener acceso a los pines de forma pasante ya que el conector es de montaje superficial. Con una tira de pines a modo de zócalo integramos esta PCB en nuestra placa. Se conecta adecuadamente este conector nuevo a los pines 1, 27 y 28 del PIC, para conformar el nuevo circuito de programación.

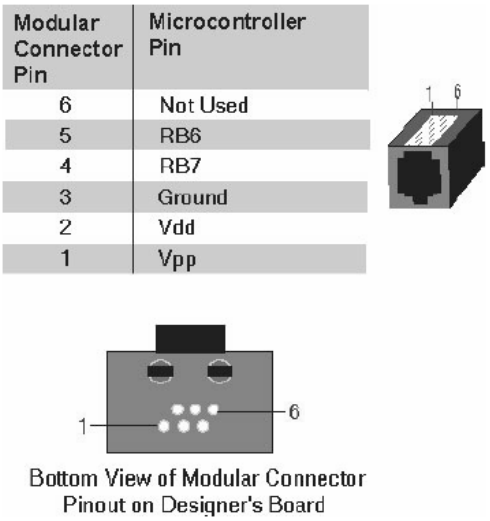


Figura 4.32: Pines conector RJ11

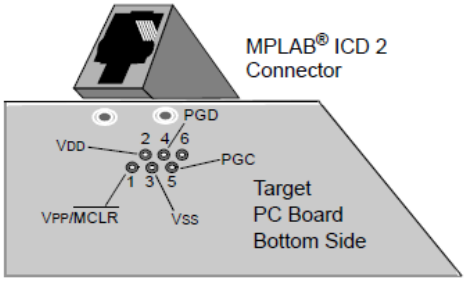


Figura 4.33: Conexión a placa del conector RJ11

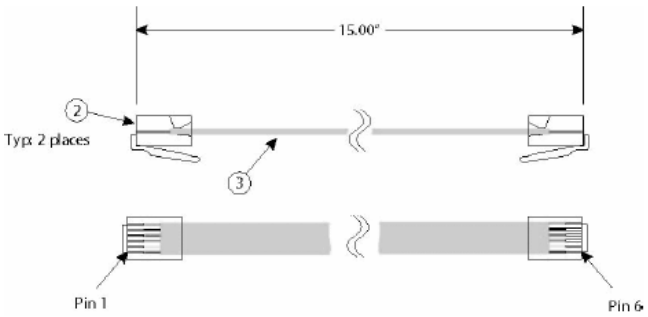


Figura 4.34: Cable del programador MPLAB ICD2 con extremos RJ11

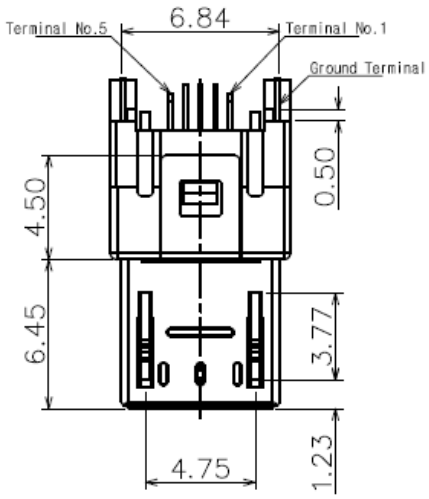


Figura 4.35: Conector microUSB macho

Como el cable de conexión del programador MPLAB ICD2 es un cable tipo telefónico con conectores RJ11, es necesario crear un adaptador personalizado para pasar de conectores RJ11 a conectores microUSB. Para ello se corta en el menor tamaño posible una placa de puntos de forma rectangular que sirva de soporte, y se sueldan en ambos extremos un conector RJ11 hembra y un conector microUSB macho. Atendiendo a los dibujos de líneas de conexión que se pueden ver en las figuras 4.32, 4.33, 4.34 y 4.35, y teniendo en cuenta que el conector RJ11 tiene 6 pines y el microUSB 5 pines, se conectan los pines de ambos conectores adecuadamente para dar como resultado el adaptador RJ11–microUSB de la figura 4.36.

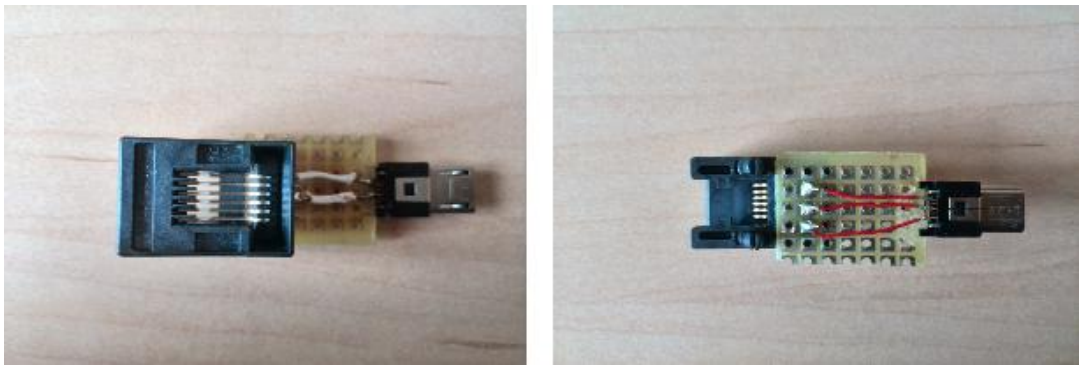


Figura 4.36: Adaptador RJ11–microUSB

Este adaptador, por un lado, engancha el extremo microUSB macho con el conector microUSB hembra instalado en la placa experimental, y por el otro lado, engancha el extremo RJ11 hembra con el conector RJ11 macho del cable del programador del MPLAB ICD 2; figura 4.37. Hay que tener en cuenta que este adaptador es indispensable para programar el PIC en la placa definitiva, ya que precisamente el acceso a éste será a través de conexión microUSB.

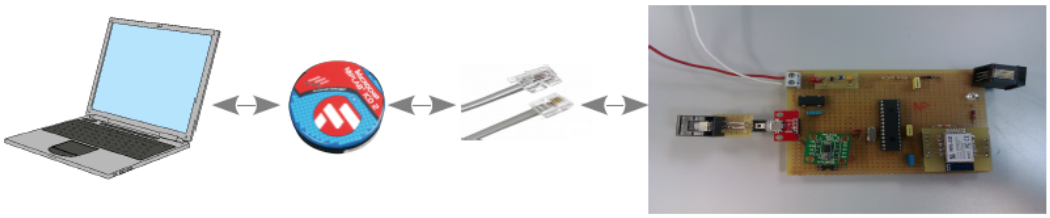


Figura 4.37: Esquema para programar con el adaptador RJ11–microUSB

4.3. Placa definitiva. Layout PCB

Primero se ha diseñado el esquemático del circuito en OrCAD Capture CIS y se ha probado su funcionalidad en la placa experimental (ver capítulo 7). Lo siguiente es transferir el circuito al programa OrCAD Layout Plus para proceder al diseño del layout de la placa de circuito impreso (PCB) definitiva [40].

En el Apéndice B se puede consultar el listado definitivo de componentes de esta placa definitiva, todos ellos de montaje superficial excepto el interruptor deslizante que es de montaje through-hole o pasante. Como la mayoría de estos componentes son muy particulares, no aparecen en las librerías de OrCAD Layout Plus, por lo que hay que crear sus huellas o footprints personalizados. Buscando en la hoja de características de cada componente la información sobre dimensiones de layout, se aplican estos datos al diseño de cada huella. Hay que tener en cuenta:

- Las dimensiones exactas de los pads de contacto de los pines (color azul en las figuras)
- El espaciado 'clearance' de seguridad que se deja alrededor del pad para indicar que ninguna pista de cobre puede pasar por encima (color gris) [nota: el color es verde en algunas figuras por estar visualizándose otra capa en ese momento]
- La línea de encapsulado que marca la vista en planta del componente, indicando el espacio exacto que va a ocupar el encapsulado del componente en la placa (capa SSTOP)(color blanco)
- El outline o línea de límite de la huella, que indica el espacio total que va a ocupar el conjunto de encapsulado y pads, y que sirve para evitar solapamientos (color verde)
- El nombre del componente, para ayudar en el montaje final (capa SSTOP)(color blanco) [nota: en realidad el nombre se coloca en la capa ASYTOP y es de color verde, pero para facilitar su visualización se duplica también en la capa SSTOP)

Hay otros parámetros que hay que configurar para una correcta creación de la huella, pero al no visualizarse en las figuras no se van a comentar. Lo que sí es importante es saber que las medidas utilizadas en diseño de layout y en general en tecnología electrónica se basan en las pulgadas, por lo que hay que tener siempre presente la relación de equivalencias de la tabla 4.6. 1 mil = 1 milésima de pulgada. El estándar de separación de las patillas de los componentes de montaje pasante es de 100 mil (2,54 mm) y en general todas las medidas de

encapsulados son múltiplos de 100 mil. En OrCAD Layout Plus se trabaja con la medida mil, por lo que hay que tenerlo en cuenta al tomar las medidas de dimensiones de las hojas de características.

Tabla 4.6: Tabla de equivalencias pulgada–mil–mm

1 pulgada	2,54 cm
1 mil	0,0254 mm
100 mil	2,54 mm

Los componentes de los que han tenido que crearse nuevas huellas o footprints son: ADXL345, PIC16LF876A, RN42, TPS62203, oscilador de cristal SMD, diodo LED SMD, interruptor deslizante, conector microUSB y portapilas botón; véanse las figuras 4.38, 4.39, 4.40, 4.41, 4.42, 4.43, 4.44, 4.45 y 4.46. Éste último, al estar situado en la cara BOTTOM, tiene una visualización distinta. También decir que las footprints del RN42 y del TPS62203 ya se crearon al realizar la placa experimental. Del resto de componentes, que son los pasivos, se ha buscado la huella adecuada en el programa, en este caso 0603 ó 0402.

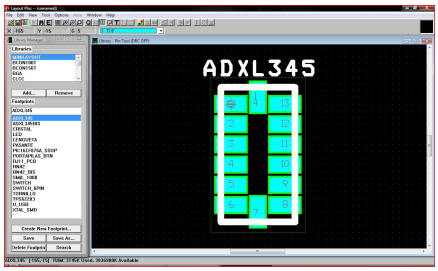


Figura 4.38: Footprint del ADXL345

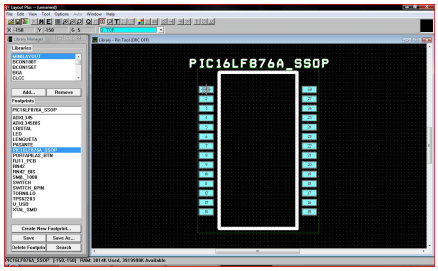


Figura 4.39: Footprint del PIC16LF876A

El diseño general de la PCB estará basado en dos características principales: la PCB va a ser circular y tiene que tener las dimensiones, es decir, el diámetro, más pequeño que se pueda conseguir. Que el dispositivo sea lo más pequeño posible es uno de los objetivos principales del proyecto y el diseño circular hace por un lado la placa más pequeña al eliminar esquinas y por otro lado hace que el dispositivo sea más manejable al ser usado por la persona.

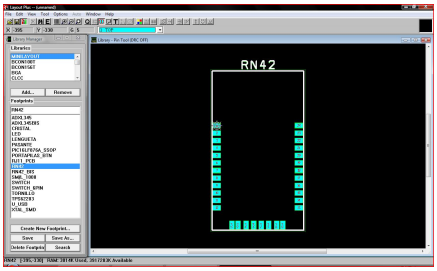


Figura 4.40: Footprint del RN42

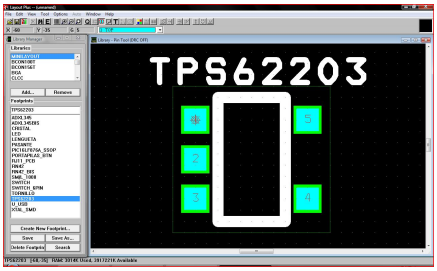


Figura 4.41: Footprint del TPS62203

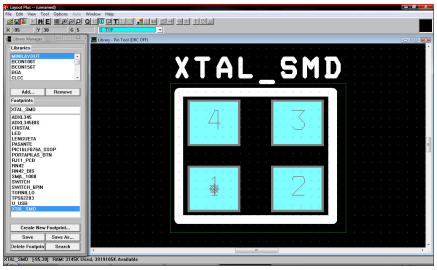


Figura 4.42: Footprint del Oscilador de Cristal SMD

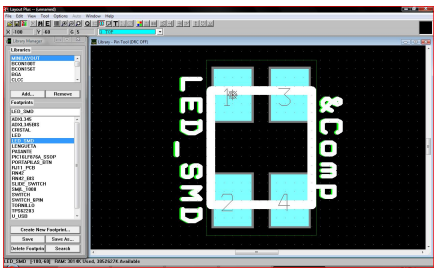


Figura 4.43: Footprint del LED SMD

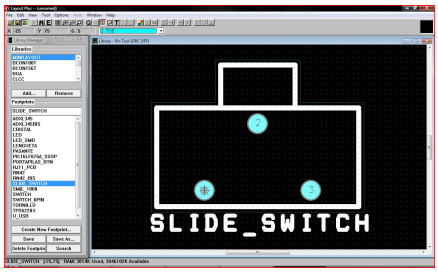


Figura 4.44: Footprint del Interruptor Deslizante

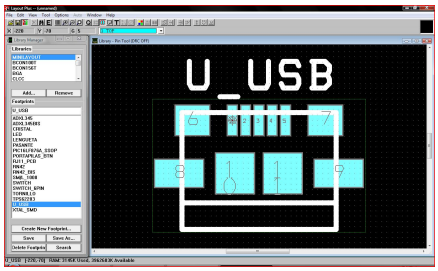


Figura 4.45: Footprint del Conector microUSB hembra

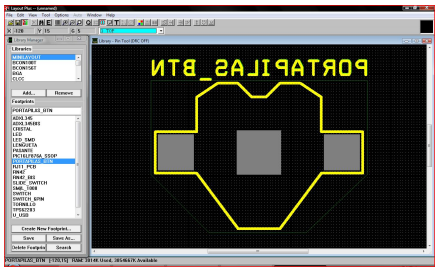


Figura 4.46: Footprint del Portapilas botón para PCB

Siguiendo estos principios, y al igual que en cualquier diseño de PCB, el secreto está en encontrar la ubicación óptima de los componentes dentro de la placa, para facilitar al máximo las conexiones y que el rutado de las pistas sea el menor y el más fácil posible. El componente limitador para el diseño es el que sea el más grande de todos, que en este caso es el RN42, con mucha diferencia de tamaño respecto de todos los demás componentes. Además, tiene la particularidad de que la antena emisora de radiofrecuencia de este módulo Bluetooth, según la hoja de características, no puede ser solapada por ninguna pista de cobre ni plano de masa, teniendo además una zona de exclusión alrededor de ella, todo esto para asegurar una emisión sin interferencias, o directamente, para poder emitir. Por ello, hay que situar el RN42 con la antena en un extremo de la PCB, apuntando hacia afuera.

Por otro lado, el otro componente de mayor tamaño es el portapilas botón, y además son 3 unidades. Esto ya se tenía previsto desde el principio, puesto que su ubicación va a estar en la cara BOTTOM. Se va a fabricar una PCB de dos caras, TOP y BOTTOM. En la cara TOP se van a situar todos los componentes y en la cara BOTTOM van a estar los 3 portapilas y las pistas de conexión extra que se necesiten por falta de espacio en la cara TOP. La idea preliminar de ubicación en la PCB al comenzarse este proyecto se puede ver en la figura 4.47. Se estima de inicio, mirando las dimensiones de los componentes, un diámetro de PCB de 40 mm. En cuanto a la dimensión vertical, no es limitante puesto que la altura de los componentes es muy pequeña (el portapilas con 6.1 mm es el componente de más altura), y en todo caso se estima una altura total de en torno a 10 mm, que es más que aceptable.

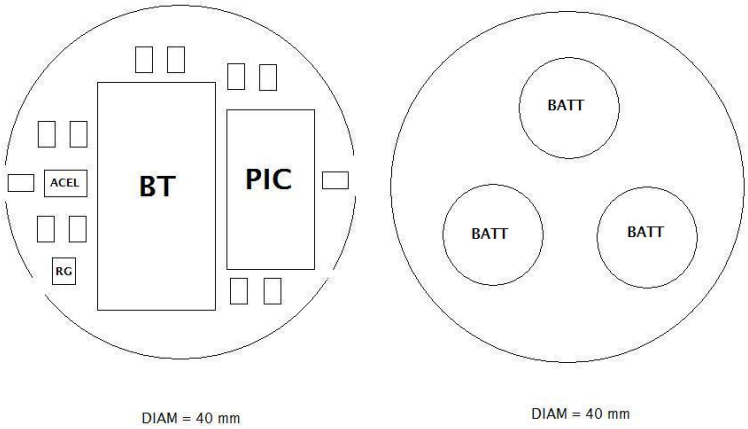


Figura 4.47: Idea preliminar de ubicación en la PCB

Otros elementos obligatorios en cuanto a la ubicación, son el interruptor deslizante y el conector microUSB hembra. Ya que la PCB irá soportada por una carcasa de plástico cubriendo todo el dispositivo, la idea es dejar las aberturas

necesarias en el lateral de la carcasa para estos dos componentes. Por tanto deberán ubicarse justo en el borde de la PCB.

El siguiente componente fundamental es el microcontrolador PIC16LF876A, ya que es el segundo componente electrónico más grande, es parte central del diseño eléctrico y va conectado a prácticamente todo el resto de elementos, y por tanto, es el componente del que más pistas de rutado saldrán. Es por ello que debería de ir ubicado lo más centrado posible.

El acelerómetro ADXL345 es pequeño, pero su requisito en cuanto a la ubicación es que su encapsulado, y por extensión, sus ejes x e y, estén alineados con el dispositivo de tal manera que visualmente desde el exterior se pueda saber la posición del acelerómetro. Esto se hace alineándolo con el interruptor, con el conector microUSB o con los LEDs, que serán visibles desde el exterior de la carcasa soporte del dispositivo.

Los LEDs de estado se deben de ver, y se van a realizar las aberturas necesarias en la carcasa para verlos, por tanto su ubicación no debe de ser arbitraria, se pretende guardar un mínimo de estética en el dispositivo final. Eso sí, lo prioritario es su correcta conexión y funcionalidad.

En cuanto al oscilador de cristal sólo tener en cuenta que tiene que estar ubicado lo más cerca posible del PIC para asegurar un correcto funcionamiento del reloj y del conjunto.

El regulador de tensión TPS62203 estará ubicado cerca del interruptor del cual vendrá la alimentación proporcionada por las pilas, y a partir de éste, se distribuirá la señal de 3.3 V que alimentará a todos los componentes del circuito.

La ubicación de los condensadores y de la inductancia tiene que ser lo más cercana posible a sus respectivos componentes para asegurar su buen funcionamiento. Sin embargo, las resistencias sí tienen flexibilidad a la hora de situarlas en la PCB. En todo caso, estos pasivos son muy pequeños por lo que van a dar muy pocos problemas.

Aún teniendo claros los requisitos de diseño, el proceso de ubicación y rutado de los componentes para llegar a la composición final de la PCB, ha sido muy laborioso, minucioso y difícil. Hay que tener en cuenta que esto ha sido así ya que el objetivo principal en este proceso es obtener las dimensiones más pequeñas posibles. Si no se hubiera reparado en el tamaño, los componentes podrían estar más espaciados y este proceso habría resultado más sencillo y rápido, a costa de tener una placa exagerada de diámetro. Pero no era el caso. Sí se puede decir que el resultado final ha merecido la pena, y sobre todo, funciona.

Una vez transferido el circuito eléctrico representado por el esquemático des-

de OrCAD Capture CIS, al programa OrCAD Layout Plus, es aquí donde se realiza el diseño de la PCB. OrCAD crea un archivo con una netlist en la que vienen codificadas todas las conexiones eléctricas de todos los componentes del circuito entre sí. Una vez asignadas las huellas o footprints correspondientes a cada elemento, se tiene una representación gráfica de todos los componentes, sus pines y las conexiones entre sí, incluyendo masa. La labor consiste en emplazar todos los componentes de acuerdo a lo expuesto anteriormente, delimitándolos dentro de una línea de límite de PCB llamada Board Outline (amarilla ancha). De todas las capas del diseño, son en las capas TOP y BOTTOM donde se realiza el rutado de las pistas de cobre. Por tanto una vez realizado un primer emplazamiento de todos los componentes, se comienza el rutado de pistas. Aunque el programa permite hacer un emplazamiento y un rutado automático, en nuestro caso no podemos utilizarlo primero porque se pretenden emplazar los componentes con los requisitos mencionados anteriormente y segundo porque el rutado automático no es capaz de solucionar el proceso ante la complejidad y el poco espacio disponible. Hay que señalar que por esta razón ya se ve desde el principio la necesidad de completar rutados en la placa BOTTOM mediante el uso de vías, que son agujeros pasantes de cobre que conectan eléctricamente la cara TOP y la cara BOTTOM. Este trabajo no lo puede hacer automáticamente OrCAD. Por ello, comienza un trabajo manual recurrente de rutado-reemplazamiento-rutado para dar solución al rutado de todo el circuito y que además se consiga con el menor diámetro de placa posible.

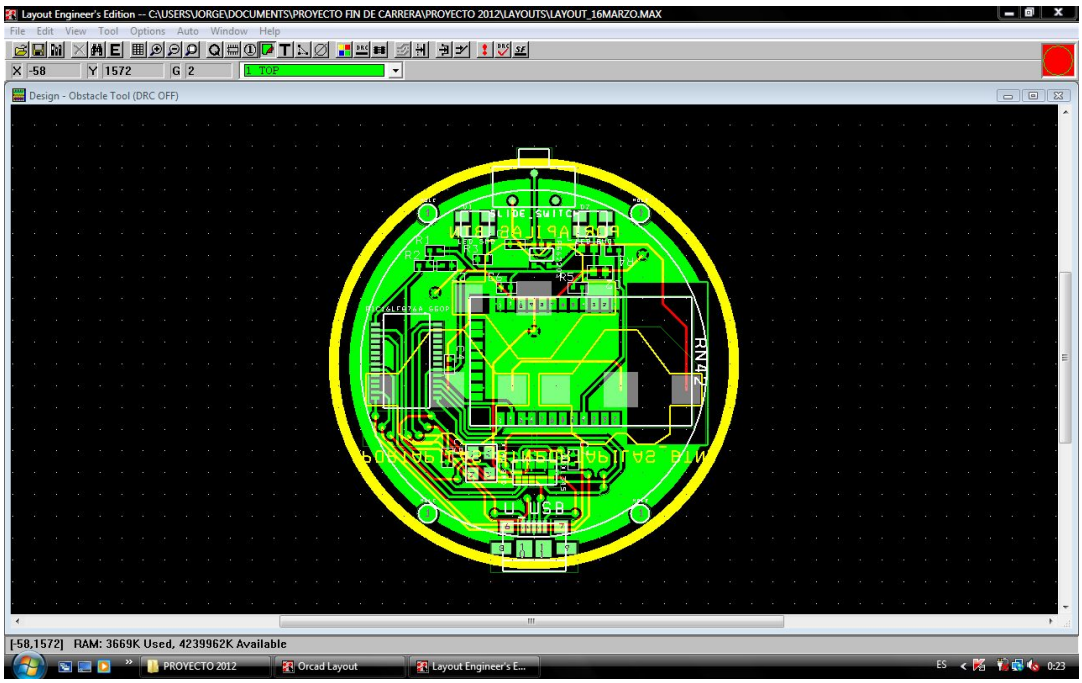


Figura 4.48: Aspecto final del layout definitivo con todas las capas en OrCAD Layout Plus

Ante la gran cantidad de conexiones a masa (0) del circuito se decide rutar

este punto eléctrico con un plano de masa, que es un plano de cobre que se extiende por toda la capa TOP. Se define un parámetro 'clearance' de plano de masa que indica la separación de este plano de masa con el resto de pistas. Ante la gran cantidad de conexiones y componentes, y el poco espacio que hay, se van a generar bastantes islas en las que el plano de masa no llega. Esto se va a solucionar con el uso de vías y rutados en la capa BOTTOM.

Otra característica a tener en cuenta es que se van a realizar 4 agujeros pasantes en la PCB para dejar pasar los tornillos de sujeción que sujetarán la PCB a la carcasa de plástico que servirá de soporte mecánico. Estos agujeros hay que incluirlos en el diseño de la placa emplazándolos en los extremos de manera simétrica.

Con todas estas consideraciones, el proceso de rutado, re-emplazamiento de componentes y re-rutado se repite numerosas veces hasta llegar a la solución final, que consigue rutar todas las conexiones eléctricas del circuito y emplazar de manera satisfactoria todos los componentes, con el menor diámetro de PCB posible, figura 4.48.

Una vez terminado el diseño del layout de la PCB, se procede a la generación de los fotolitos que representan el aspecto final de cada una de las capas que se necesitan para la fabricación de la placa.

Las capas que se han utilizado son: TOP, BOTTOM, INNER, PLANE, SMTOP, SMBOT, SSTOP, SSBOT, ASYTOP, ASYBOT, DRLDWG y DRILL. En el Apéndice D se pueden ver los fotolitos de las capas más importantes.

La fabricación de la tarjeta PCB se ha realizado en la Universidad con coste cero. Sin embargo, para el montaje de los componentes en la PCB se ha recurrido a una empresa externa a la Universidad, Madritronic S.L., ya que debido al pequeño tamaño de la placa y los componentes de montaje superficial (SMD), se requiere de tecnología y maquinaria de los que la Universidad no dispone. Para ello, y a través de la Oficina Técnica de la Universidad, se efectúa una petición de trabajos externa, en la que se piden los siguientes archivos:

- Ficheros gerber (generados a partir del diseño final en OrCAD Layout Plus)

- Archivo .max (archivo principal de OrCAD Layout Plus con todo el diseño y parámetros)

- Listado de componentes definitivo

- Esquemático definitivo

Hay que señalar que las baterías tipo botón, obviamente, no se incluyen aquí. En la PCB van montados los portapilas, las pilas se utilizarán ya como usuario en la puesta en funcionamiento. Señalar también que la programación del dispositivo definitivo y las consiguientes pruebas de funcionamiento finales se realizan, obviamente, a la entrega de la fabricación y montaje de la PCB.

Una vez explicado todo el proceso de diseño de la PCB, el aspecto final de la placa definitiva se puede ver en las figuras 4.49, 4.50, 4.51, 4.52, 4.53 y 4.54.

Figura 4.49: PCB definitiva. Referencia de tamaño

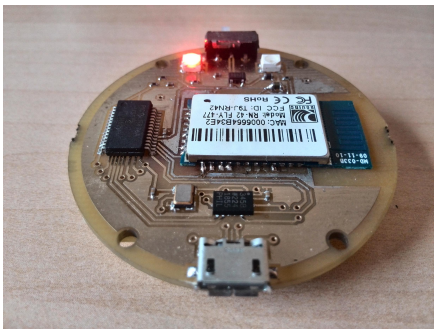


Figura 4.50: PCB definitiva. Posición ON. LED rojo

Por último, en este punto ya se dispone de las dimensiones definitivas del dispositivo, del peso total, del consumo total con todos los componentes definitivos y también de la autonomía de funcionamiento.

- Dimensiones definitivas

Figura 4.51: PCB definitiva. Posición ON. LEDs rojo y verde

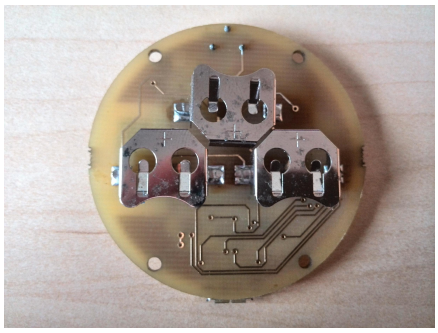


Figura 4.52: PCB definitiva. Vista desde abajo en planta

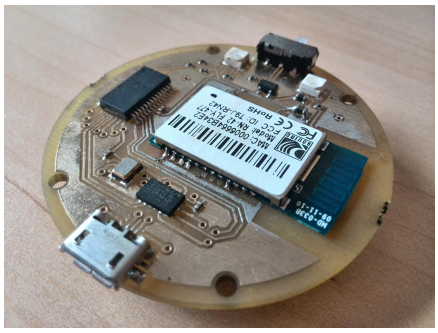


Figura 4.53: PCB definitiva. Diagonal desde conector microUSB hembra

Figura 4.54: PCB definitiva. Diagonal desde interruptor deslizante

Del diseño del layout se obtiene el diámetro final de la PCB, que es de 1800 mils = 45,72 mm = 4,572 cm.

La altura de la placa PCB es de 1 mm (medida estándar).

La altura final con los componentes montados es de 10,6 mm = 1,06 cm; ecuación (4.4).

- Altura PCB = 1 mm
- Altura en cara TOP: interruptor = 3,5 mm [Nota: según hoja de características el componente más alto sería el LED con un valor de 3,87 mm (max); al recibir la PCB montada se ve que no llega a ese máximo, siendo el componente de más altura el interruptor]
- Altura en cara BOTTOM: portapilas = 6,1 mm

$$1 + 3{,}5 + 6{,}1 = 10{,}6\,\text{mm} \tag{4.4}$$

- Peso total

  El peso total de la PCB con los componentes montados sin pilas es de 9 gramos.

  El peso total de la PCB con los componentes montados y con pilas incluidas es de 15 gramos.

- Consumo total

  En el cálculo del consumo total del dispositivo, se van a considerar los consumos máximos de cada componente, según hojas de características.

$$i_{ADXL345} = 140\,\mu\text{A} = 0{,}14\,\text{mA}$$

$$i_{PIC16LF876A} = 2\,\text{mA}$$

$$i_{RN42} = 50\,\text{mA}$$

$$i_{TPS62203} = 0{,}1\,\mu\text{A} = 0{,}0001\,\text{mA} \rightarrow \text{despreciable}$$

$$i_{XTAL} = 1{,}5\,\text{mA}$$

$$i_{R\_LEDs} = i_{R\_LEDrojo} + i_{R\_LEDverde} = 5\,\text{mA} + 10\,\text{mA} = 15\,\text{mA}$$

(explicado en el capítulo 4.1.2)

$$i_{R\_program} = 0{,}33\,\text{mA} \text{ (resistencia de 10 k; Vcc/10k=3.3/10k=0.33 mA)}$$

$$i_{TOTAL} = i_{ADXL345} + i_{PIC16LF876A} + i_{RN42} + i_{XTAL} + i_{R\_LEDs} + i_{R\_program} = 0{,}14 + 2 + 50 + 1{,}5 + 15 + 0{,}33 = 68{,}97\,\text{mA} \quad (4.5)$$

Por tanto, según la ecuación (4.5), el consumo total máximo es de 68,97 mA.

- Autonomía

  La autonomía del dispositivo, a raíz del consumo total, y teniendo en cuenta que la capacidad de las baterías es de 165 mAh, es de 2,39 horas. Ver ecuación (4.6).

$$\frac{\text{Capacidad Baterías}}{\text{Consumo Total}} = \frac{165\,\text{mAh}}{68{,}97\,\text{mA}} = 2{,}39\,\text{h} \rightarrow 2 \text{ horas y } 23.5 \text{ minutos} \quad (4.6)$$

La tabla 4.7 resume las características principales del dispositivo definitivo, cuyo aspecto final se vio en las figuras 4.49, 4.50, 4.51, 4.52, 4.53 y 4.54.

Tabla 4.7: Resumen características principales del dispositivo definitivo

Característica	Valor
Diámetro	4,572 cm
Altura total	1,06 cm
Peso total sin pilas	9 gramos
Peso total (pilas incluidas)	15 gramos
Consumo total	68,97 mA
Autonomía	2,39 horas

Capítulo 5

Diseño Software

El software de la interfaz tiene dos vertientes. Por un lado hay que programar el microcontrolador PIC del dispositivo para que lea las medidas provenientes del acelerómetro, haga un tratamiento adecuado de esa información, y la transmita al módulo Bluetooth, que será el encargado de enviar los datos de forma inalámbrica al ordenador. Por otro lado, se crea un programa residente en el ordenador que sea capaz de conectar con el módulo Bluetooth del dispositivo, recibir la información y procesarla adecuadamente para la finalidad deseada, que en último caso es transmitir el estado real de inclinación de cada eje del dispositivo en cada momento, al robot. En esta fase de desarrollo estos valores se muestran en pantalla, ya que el algoritmo de paso de los valores al robot es cuestión de desarrollos futuros, lo cual se explica en el capítulo 8.

El proceso de programación del PIC se realiza a través de un entorno completo de programación del fabricante, Microchip, que es el programa MPLAB IDE [41]; figura 5.1. Este entorno informático permite crear código fuente, permite ensamblar, compilar y 'linkear' ese código, permite depurar el programa mediante visión de las variables en ventanas o mediante simulación, y permite crear un archivo ejecutable y programarlo en el dispositivo seleccionado a través de un programador específico. La programación se va a realizar en lenguaje C.

En este caso, se elige para efectuar la creación y edición de código el programa PIC C Compiler (figura 5.2), ya que MPLAB IDE soporta herramientas CCS, que son editores de código y compiladores externos para lenguaje C. PIC C Compiler es un editor más potente que el integrado en MPLAB, sobre todo, facilita mucho la sintaxis y la detección de errores de compilación [42]. Por ello, todo el trabajo de edición de código se realiza con este programa, y una vez terminado y comprobado, se pasa al entorno de MPLAB IDE para probar su funcionalidad [43].

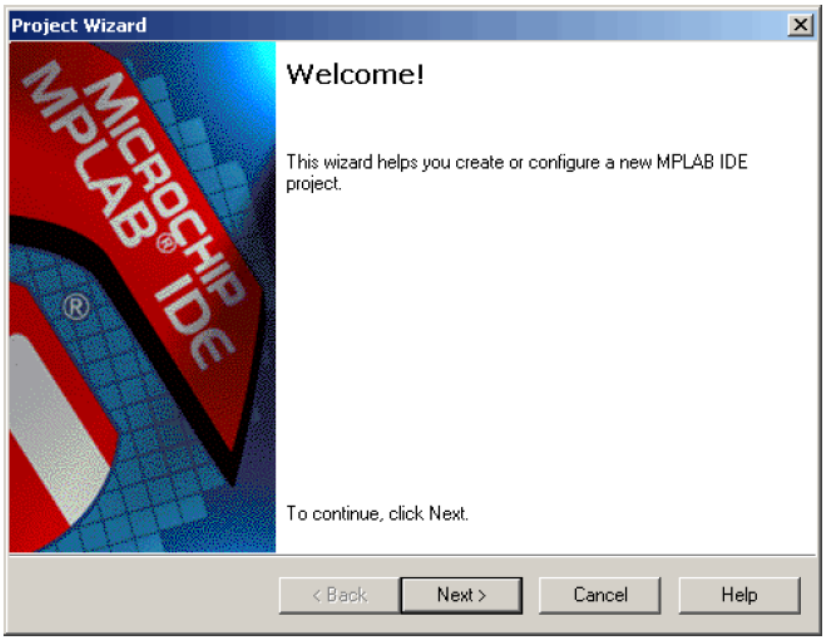


Figura 5.1: Pantalla de bienvenida del entorno informático MPLAB IDE

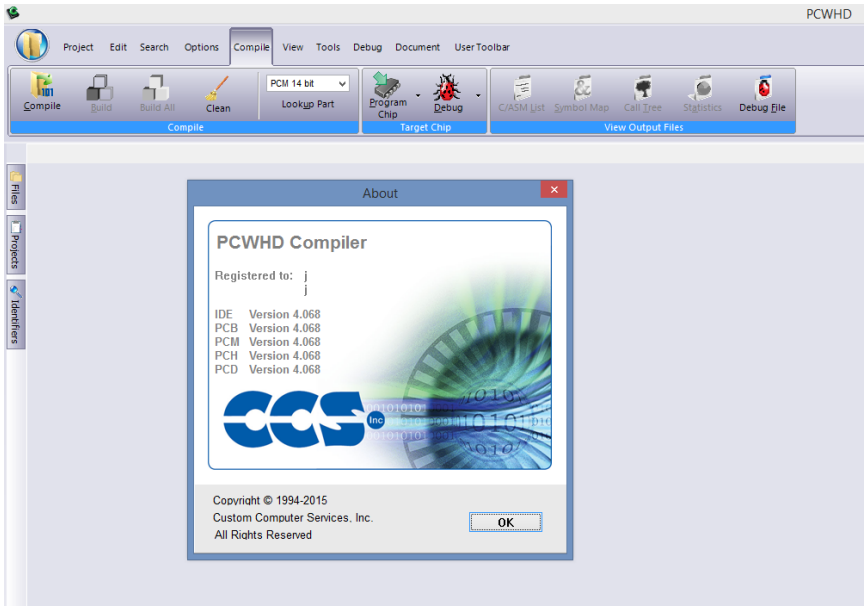


Figura 5.2: Pantalla de PIC C Compiler

El programador utilizado para programar el PIC es el MPLAB ICD2; figura 5.3. Por un lado se conecta al ordenador mediante toma USB y por el otro lado se conecta a la placa donde esté montado el PIC mediante conectores RJ11. Este programador se selecciona desde MPLAB IDE y puede realizar dos funciones: In-Circuit Debugger, depurador, ó bien, In-Circuit Programmer, programador. Se basa en la tecnología In-Circuit Serial Programming (ICSP) de Microchip, que lo que permite es depurar o programar el PIC directamente montado en la placa, o dicho de otra forma, permite depurar o programar sistemas embebidos basados en microcontroladores PIC [44]. Al depurar, se programa en el PIC un archivo de depuración equivalente al definitivo para efectuar todas las pruebas necesarias estando conectado necesariamente el dispositivo al ordenador a través del programador. Así, se puede hacer un seguimiento del funcionamiento del programa en MPLAB IDE. Al programar, lo que se programa en el PIC es un archivo ejecutable definitivo para que funcione ya de forma autónoma sin estar conectado al ordenador.

Figura 5.3: Programador MPLAB ICD2

El desarrollo de creación del software para la interfaz va a tener los siguientes pasos. En primer lugar se van a escribir varios códigos que permitan asegurar la correcta comunicación del PIC con el acelerómetro. Una vez comprobado este punto se va a escribir un programa que comunique al acelerómetro y al microcontrolador de manera funcional, leyendo medidas reales e interpretando esos datos con un sentido físico. Después se va a probar la comunicación entre el PIC y el módulo Bluetooth. Todos estos códigos se realizan de manera secuencial para ir añadiendo funcionalidades nuevas según se hayan comprobado y contrastado las anteriores. Son códigos intermedios necesarios para llegar a desarrollar el código final, por ello a su conjunto se va a denominar programación previa.

En este punto ya se puede escribir un código que relacione todos los elementos de hardware. Para comprobar el funcionamiento completo se escribe un programa que convierta el movimiento del dispositivo en los ejes x e y, a movi-

miento del cursor del ordenador (PC) en la pantalla. Aquí ya se involucra a toda la interfaz puesto que interviene el ordenador, por tanto, hay que escribir dos programas. Un programa para el microcontrolador, que basado en toda la programación previa, sea capaz de leer el movimiento del acelerómetro y mandar los datos por el módulo Bluetooth. Y otro programa, para el ordenador, que sea capaz de recibir esa información del dispositivo y traducirla a movimiento del cursor en pantalla. Este programa es un cliente del PC puesto que es ejecutado en el ordenador y se queda a la espera de conexión con el módulo Bluetooth del dispositivo, así como también se queda a la espera de los datos de movimiento que transmita éste.

Por último, la programación definitiva consiste en llevar la información del movimiento de cada uno de los 3 ejes del acelerómetro en el dispositivo, al ordenador, y que éste presente esa información de los valores de inclinación en pantalla. Ya se ha dicho que en un futuro desarrollo estos valores son transmitidos en tiempo real al robot para que éste se mueva de acuerdo con el movimiento del dispositivo. Aquí, igualmente, dos programas, uno para el microcontrolador y otro como cliente del PC.

Señalar que el programa adx_pic.c es la base de todas las versiones posteriores del programa del microcontrolador, incluidas las definitivas. En el capítulo 5.1.2 se explica de manera detallada el código, por tanto es aquí donde se puede consultar la explicación de las funciones y directivas principales que sirven como base de todos los códigos.

Las pruebas, ensayos y resultados se pueden ver en el capítulo 7.

5.1. Programación previa

5.1.1. Comunicación con el acelerómetro

Programa: comunic_id.c

Objetivo: leer por SPI el registro identificador (DEVID) del ADXL.

Planteamiento: en la figura 5.4 se puede ver el diagrama de flujo del programa. Primero se configura el PIC adecuadamente, y después se lee el registro identificador del acelerómetro.

Funciones nuevas:

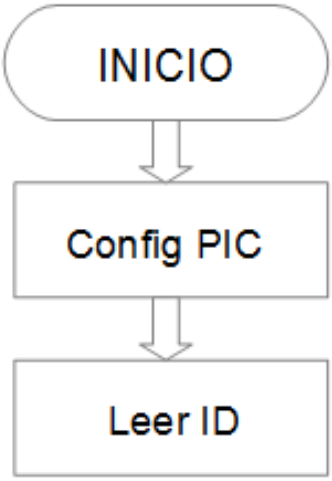


Figura 5.4: Diagrama de flujo del programa comunic_id.c

- config_PIC(): configuración del PIC; pines entrada/salida, configuración SPI.
- readADX(): lectura genérica del acelerómetro por SPI.
- lectura_id(): lectura del registro DEVID y visualización en pantalla.

Programa: comunic_id.h

Cabecera de comunic_id.c. Definición de los registros del ADXL345.

Programa: config_reg.c

Objetivo: escribir y leer por SPI todos los registros del ADXL.

Planteamiento: después de configurar el PIC, se configura el acelerómetro. Tras ello se hace una lectura de todos los registros del ADXL. Figura 5.5.

Funciones nuevas:

- readADX(): se modifica del anterior código para poder leer múltiples bytes.
- writeADX(): escritura genérica en el acelerómetro por SPI.

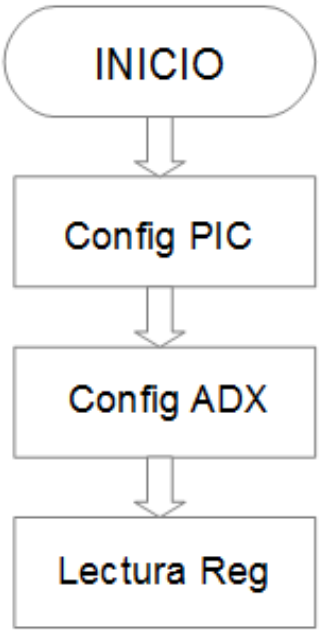


Figura 5.5: Diagrama de flujo del programa config_reg.c

- config_ADX(): escritura de registros del ADXL.
- lectura_reg(): lectura de registros del ADXL.

Programa: config_reg.h

Cabecera de config_reg.c. Definición de los registros del ADXL345.

Programa: basic_measure.c

Objetivo: activar modo Measure, leer los datos de los ejes x, y, z, reconstruir datos.

Planteamiento: tras configurar el PIC y el ADXL, se activa el modo Measure para habilitar la medición de valores de inclinación por parte del acelerómetro. Posteriormente se hace una lectura continua de esos valores. Figura 5.6.

Funciones nuevas:

- lectura_ejes(): lectura de los registros de los ejes x, y, z del ADXL, y re-

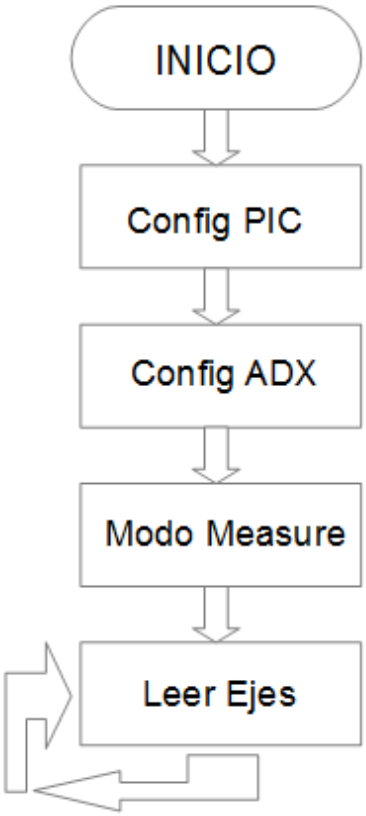


Figura 5.6: Diagrama de flujo del programa basic_measure.c

construcción de los datos para formar la palabra de 10 bits de cada eje.

Programa: basic_measure.h

Cabecera de basic_measure.c. Definición de los registros del ADXL345.

El funcionamiento físico del acelerómetro ADXL345 hace depender la medida de cada eje en función de la posición que adopta el sensor respecto a la vertical, respecto a la gravedad. En la figura 5.7 se puede ver lo que se considera posición de reposo del acelerómetro, y en la figura 5.8 se ve cómo la medida de cada eje se ve afectada según la posición espacial del sensor.

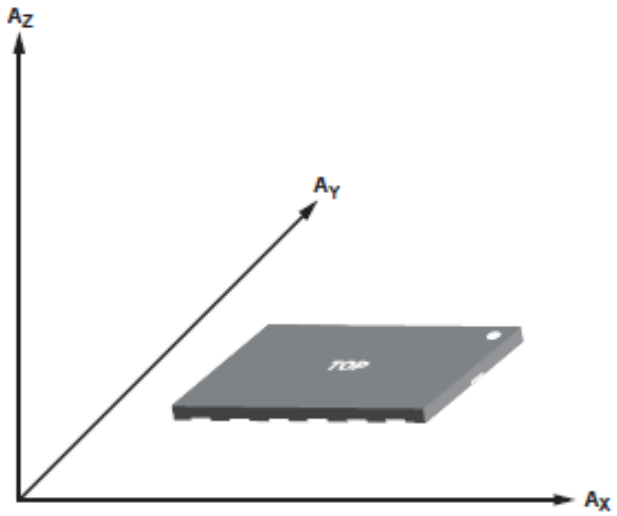


Figura 5.7: Posición de reposo del acelerómetro

5.1.2. Programación acelerómetro–microcontrolador

Programa adx_pic.c

Objetivo:

Establecer una lectura inicial de datos para ver el rango de valores de cada eje en la posición inicial. Ajustar los registros de offset del ADXL teniendo en cuenta los valores iniciales, este ajuste hace que la salida de datos en cada eje esté corregida según el offset inicial. Realizar lecturas continuas de datos e interpretarlos según su significado en movimiento real.

Planteamiento:

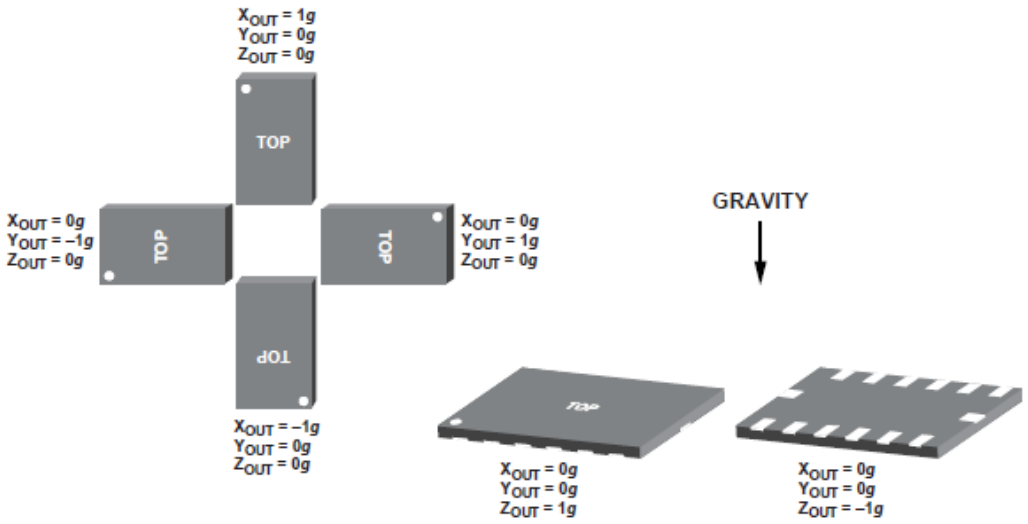


Figura 5.8: Respuesta de cada eje del acelerómetro según posición espacial

Se van a establecer tres zonas de funcionamiento, una zona de reposo, una zona de movimiento normal y una zona de movimiento rápido; ver figura 5.9. La zona de reposo se crea debido a la gran sensibilidad del acelerómetro que aún estando en reposo mide pequeños valores en cada eje, y se trata de que en reposo no se haga ninguna acción. La zona de movimiento rápido se crea para que ante inclinaciones muy grandes del dispositivo, el movimiento sea más rápido y así tener mejor respuesta en el funcionamiento. Se establecerán unos valores umbrales r, v para separar cada zona de funcionamiento. Para discernir entre los distintos movimientos arriba, abajo, derecha e izquierda, se utilizan las rectas $y = 2x$, y la recta $y = 0{,}5x$ como separadores de zonas y así ser más realistas y precisos en los movimientos combinados. Combinando esta idea con la anterior se tiene la figura 5.10, en la que se pueden ver todas las zonas de funcionamiento del dispositivo. En el diagrama de flujo de las figuras 5.11 y 5.12 se puede ver la secuencia de acciones que determinan la estructura del programa.

Funciones nuevas:

- lectura_inic(): se toman 10 muestras por cada eje de la posición inicial y se hace la media.
- ajuste_offs(): ajuste de offsets en cada eje según hoja de características.
- lectura_actual(): se realiza todo el proceso de adquisición de datos e interpretación de los mismos según la realidad física.

A continuación se va a explicar con detalle el código en su totalidad, incluido el código cabecera que es adx_pic.h.

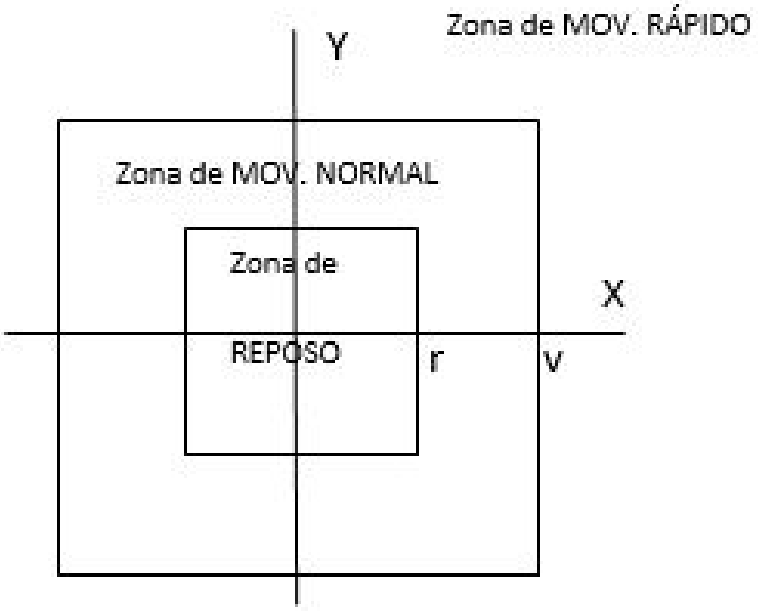


Figura 5.9: Zonas de funcionamiento con valores umbrales genéricos

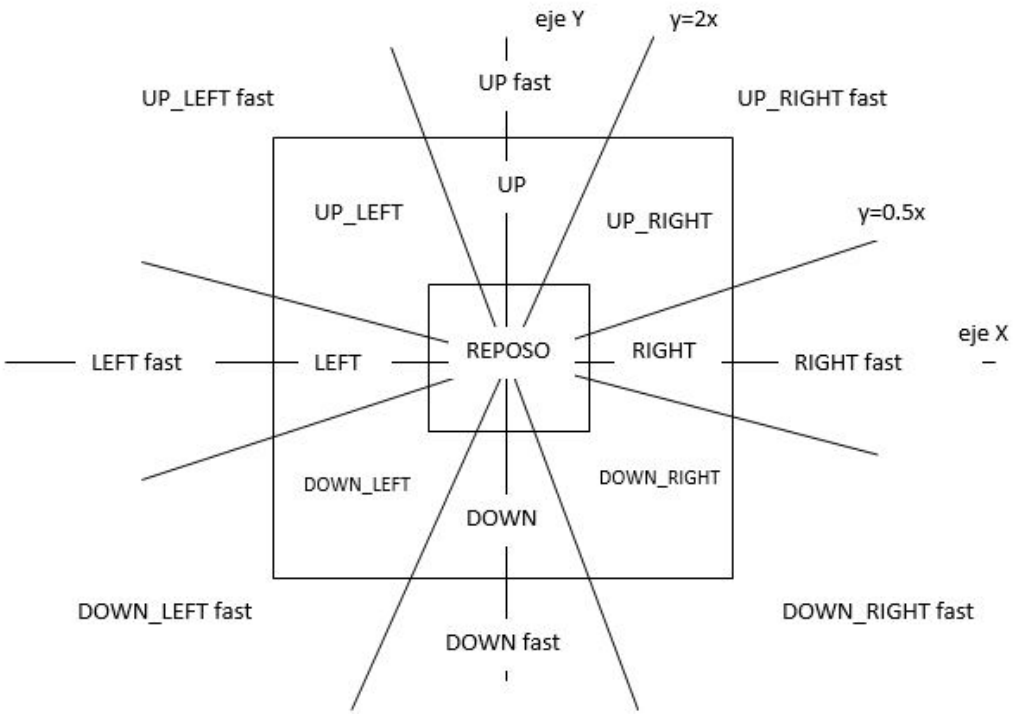


Figura 5.10: Zonas de funcionamiento. Esquema completo

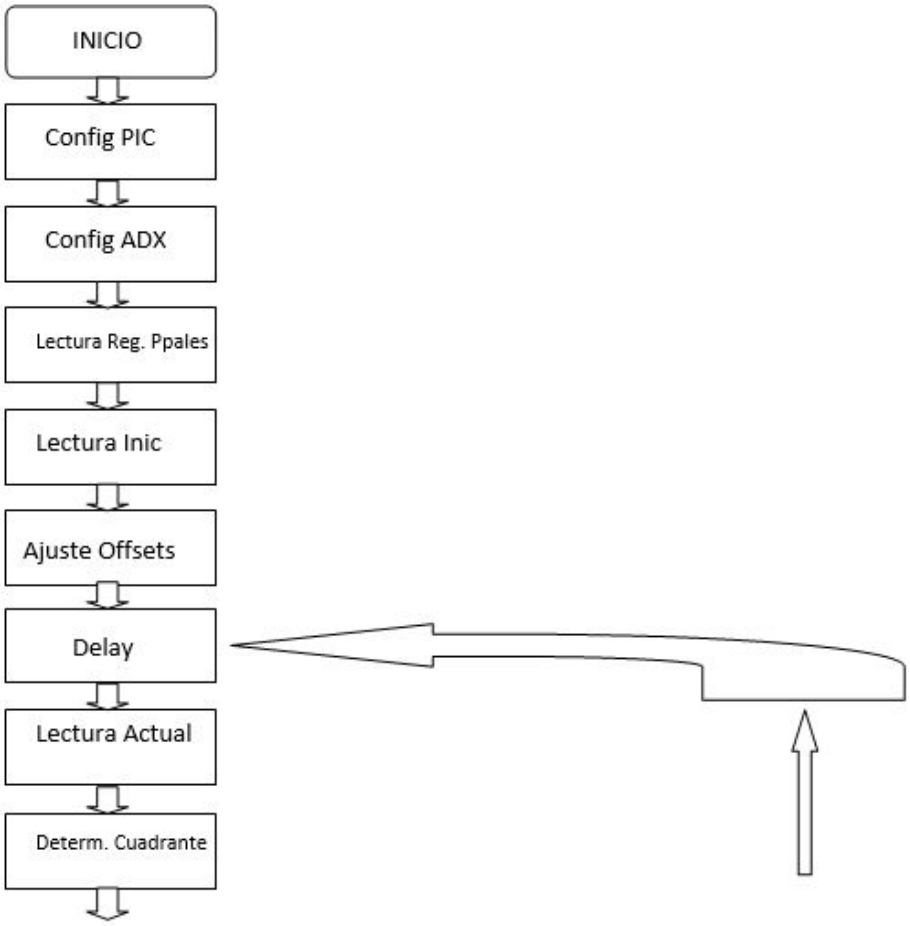


Figura 5.11: Diagrama de flujo completo. Parte 1

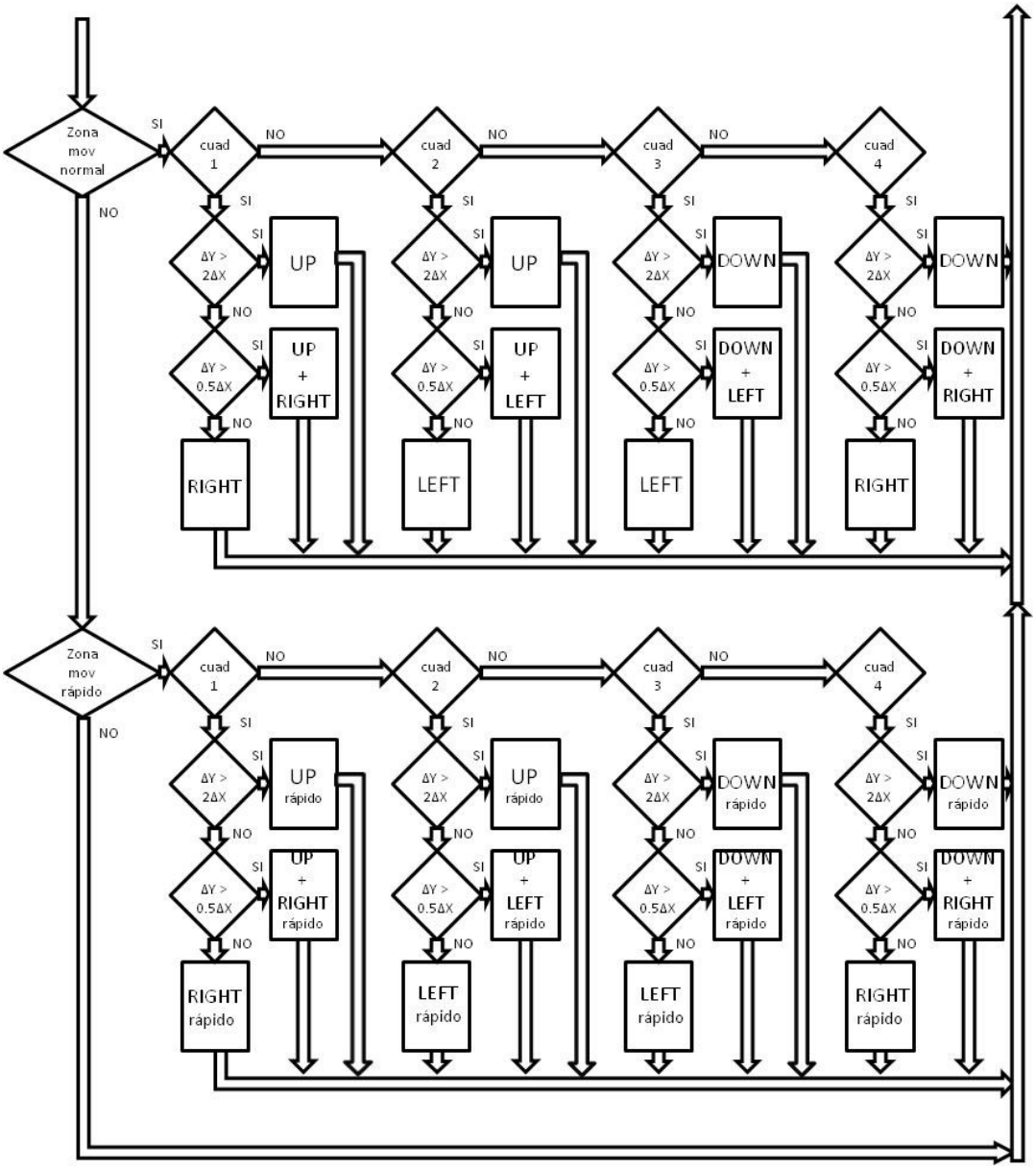


Figura 5.12: Diagrama de flujo completo. Parte 2

Explicacion detallada código adx_pic.h

Librerías que definen el dispositivo PIC concreto a utilizar, permiten el manejo de cadenas de caracteres y funciones matemáticas complejas.

```
#include < 16F876 . h >
#include < stdio . h >
#include < math . h >
```

Configuración del PIC: XT-cristal de baja frecuencia; DEBUG-permite depuración; NOLVP-pinB3 como entrada/salida-no usado para programar a bajo voltaje; el resto deshabilitan características como el watchdog-timer, power-up-time, brown-out-reset y deshabilitan protección contra escritura.

```
#fuses XT , DEBUG , NOLVP , NOWDT , NOPUT , NOPROTECT ,
        NOCPD , NOWRT , NOBROWNOUT
```

Directivas que establecen la frecuencia de reloj a 4 MHz y configuran el puerto serie USART, así como permitir usar la función printf().

```
#use delay ( clock = 4000000 )
#use rs232 ( baud = 9600 , parity = N , xmit = PIN_C6 ,
        rcv = PIN_C7 )
```

Define los valores de los registros del ADXL.

```
/* ADXL345 Register Addresses */
#define DEVID 0x00 //Device ID Register
#define THRESH_TAP 0x1D //Tap Threshold
#define OFSX 0x1E //X-axis offset
#define OFSY 0x1F //Y-axis offset
#define OFSZ 0x20 //Z-axis offset
#define DUR 0x21 //Tap Duration
#define LATENT 0x22 //Tap latency
#define WINDOW 0x23 //Tap window
#define THRESH_ACT 0x24 //Activity Threshold
#define THRESH_INACT 0x25 //Inactivity Threshold
#define TIME_INACT 0x26 //Inactivity Time
#define ACT_INACT_CTL 0x27 //Axis enable ctrl activity
        /inactiv detection
#define THRESH_FF 0x28 //free-fall threshold
#define TIME_FF 0x29 //Free-Fall Time
#define TAP_AXES 0x2A //Axis control for tap/double tap
#define ACT_TAP_STATUS 0x2B //Source of tap/double tap
#define BW_RATE 0x2C //Data rate and power mode control
```

```
#define POWER_CTL 0x2D //Power Control Register
#define INT_ENABLE 0x2E //Interrupt Enable Control
#define INT_MAP 0x2F //Interrupt Mapping Control
#define INT_SOURCE 0x30 //Source of interrupts
#define DATA_FORMAT 0x31 //Data format control
#define DATAX0 0x32 //X-Axis Data 0
#define DATAX1 0x33 //X-Axis Data 1
#define DATAY0 0x34 //Y-Axis Data 0
#define DATAY1 0x35 //Y-Axis Data 1
#define DATAZ0 0x36 //Z-Axis Data 0
#define DATAZ1 0x37 //Z-Axis Data 1
#define FIFO_CTL 0x38 //FIFO control
#define FIFO_STATUS 0x39 //FIFO status
```

Explicacion detallada código adx_pic.c

Define las constantes del usuario.

```
#include "adx_pic.h"
/* Declaracion de Constantes */
#define CSneg PIN_C2
#define user_delay 5
#define sensitivity_z 256
#define umbral_reposo 30
#define umbral_velocidad 200
```

Declara e inicializa las variables a utilizar.

```
/* Declaracion de Variables globales */
int datos [ 6 ];
int jdevid = 0 , jbw_rate = 0 , jpower_ctl = 0 ,
        jdata_format = 0 ;
signed int16 eje_x = 0 , eje_y = 0 , eje_z = 0 ,
        eje_x_real = 0 , eje_x_abs = 0 , eje_y_abs = 0 ;
signed int16 lect_x [ 10 ], lect_y [ 10 ], lect_z [ 10 ];
signed int16 suma_x = 0 , suma_y = 0 , suma_z = 0 ;
signed int16 media_inic_x = 0 , media_inic_y = 0 ,
        media_inic_z = 0 , z_cerog = 0 ;
signed int ajuste_x = 0 , ajuste_y = 0 , ajuste_z = 0 ;
int cuadrante = 0 , flag = 0 ;
char accion [ 15 ]= "--------------" ;
```

Función que configura los pines del puerto C (TRIS_C), pone a 1 el pin CSneg necesario para el acelerómetro en la configuración, y configura el puerto

serie SPI: PIC como master, flanco de bajada, velocidad de comunicación SPI de 1 Mhz, muestreo al final.

```
/* Declaracion de Funciones */
//Configuracion PIC
void config_PIC ( void )
{
set_tris_c ( 0x93 ); //pines c6,c5,c3,c2 como salidas,
        resto como entradas
output_high ( CSneg );
setup_spi ( SPI_MASTER | SPI_H_TO_L | SPI_CLK_DIV_4 |
        SPI_SAMPLE_AT_END );
//configuracion spi:master,flanco de bajada,veloc:1Mhz,
        muestra al final
}
```

Función de lectura de cualquier registro en el ADXL. Tiene como parámetros una dirección de registro, el número de bytes que se quieren leer seguidos y un puntero al array donde se guardarán los datos. Pone el bit de lectura/escritura a 1 para leer. Si se quiere leer más de un registro seguido, es decir, el número de bytes >1, se pone el Multiplebyte a 1. Se habilita la comunicación poniendo a 0 el pin CSneg, se envía el byte con la dirección de registro a leer, se lee el byte o bytes correspondientes y se deshabilita la comunicación poniendo a 1 el pin CSneg.

```
//Lectura SPI (con multipleBytes)
void readADX ( int address , int numB , int * datos )
//parametros:direccion registro adx,numero bytes(minimo2),
        puntero al array datos
{
int highbyte , i ;
highbyte = 0x80 | address ;
//pone bit lectura/escritura a 1 para leer y concatena con
        direcc de registro
if ( numB > 1 ) {
highbyte = 0x40 | highbyte ;
//si se lee mas de 1 byte seguido,se pone a 1 el bit
        multiplebyte
}
output_low ( CSneg ); //habilita comunicacion
spi_write ( highbyte ); //envia byte alto
for ( i = 0 ; i < numB ; i ++)
{
datos [ i ]= spi_read ( 0 );
//lee byte a byte y almacena en el array datos
```

```
}
output_high ( CSneg ); //deshabilita comunicacion
}
```

Función de escritura en cualquier registro del ADXL. Tiene como parámetros una dirección de registro y el valor a escribir. Se habilita la comunicación poniendo a 0 el pin CSneg, se envía el byte con la dirección de registro a leer (bit de lectura/escritura a 0 y Multiplebyte a 0), se escribe el registro con el valor correspondiente y se deshabilita la comunicación poniendo a 1 el pin CSneg.

```
//Escritura SPI
void writeADX ( int address , int valor )
//parametros: direccion registro adx, valor a escribir
{
output_low ( CSneg ); //habilita comunicacion
spi_write ( address );
//envia byte alto: bit escritura 0, bit multiplebyte 0,
        direccion registro
spi_write ( valor ); //envia valor a escribir
output_high ( CSneg ); //deshabilita comunicacion
}
```

En la figura 5.13 se puede ver el funcionamiento de la comunicación SPI con el ADXL. Es importante decir que los pulsos del reloj SCLK los generan las funciones spi_read() y spi_write(), generando 8 pulsos de reloj cada vez, asociado a un byte.

Configuración del ADXL en el que se escriben en los registros los valores deseados para configurar el sensor. Se establece comunicación SPI a 4 hilos, con resolución de 10 bits, rango de medida +/-2g, frecuencia de datos 800 Hz (máxima frecuencia recomendada para una comunicación SPI de 1 MHz). La salida de datos está justificada a derechas (LSB–byte 0), que corresponde con el formato Little–Endian. Se deshabilitan todas las interrupciones y en último lugar se pone el sensor en modo Measure para registrar las medidas en los ejes x, y, z.

```
//Configuracion ADXL345
void config_ADX ( void )
{
writeADX ( OFSX , 0x00 ); //Offset eje x = 0
writeADX ( OFSY , 0x00 ); //Offset eje y = 0
writeADX ( OFSZ , 0x00 ); //Offset eje z = 0
writeADX ( THRESH_TAP , 0x30 ); //Umbral tap 3 g
writeADX ( DUR , 0x30 ); //Duracion max tap 30 ms
writeADX ( LATENT , 0x18 ); //Espera desde tap hasta
```

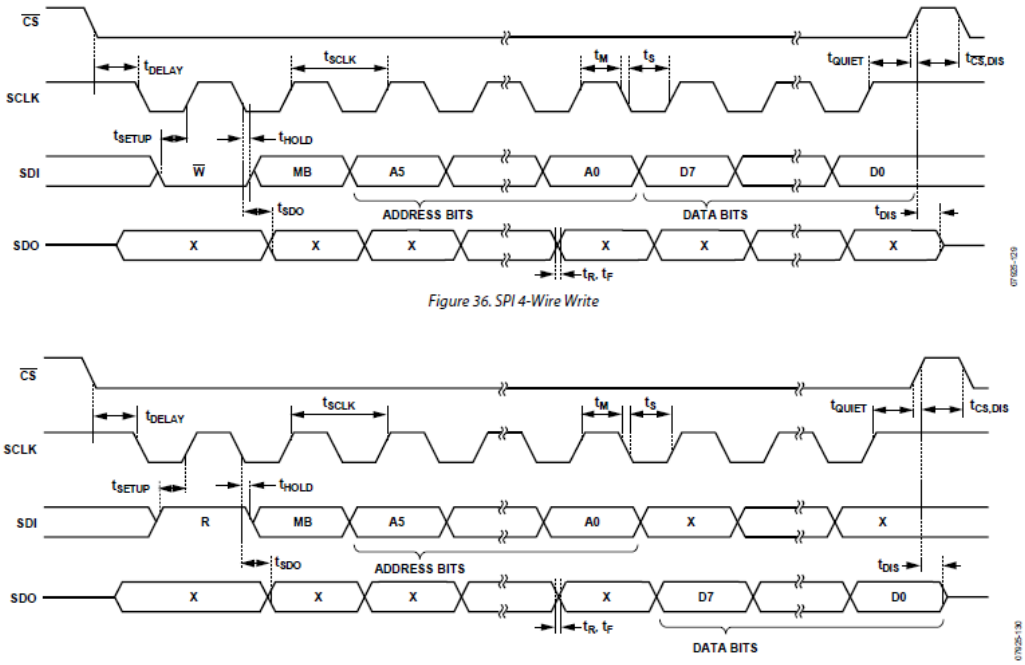


Figure 36. SPI 4-Wire Write

Figure 37. SPI 4-Wire Read

Figura 5.13: Comunicación SPI a 4 hilos

```
        window2tap 30 ms
writeADX ( WINDOW , 0x40 ); //Duracion window2tap 80 ms
writeADX ( TAP_AXES , 0x01 ); //Habilitado solo tap eje z
writeADX ( INT_MAP , 0x60 ); //Data_ready en INT1, Tap
        y 2Tap en INT2
writeADX ( BW_RATE , 0x0D ); //Data rate 800 Hz
        (pues spi_rate=1Mhz)
writeADX ( DATA_FORMAT , 0x00 ); //4 wire SPI, 10 bits,
        right justified, +/- 2g
writeADX ( INT_ENABLE , 0x00 ); //Deshabilitadas
        todas las interrupciones
writeADX ( POWER_CTL , 0x08 ); //Modo Measure
}
```

Función que lee los 6 registros seguidos de datos de los ejes x, y, z. Cada eje es una palabra de 16 bits formada por dos registros de un byte cada uno. La salida del eje x está formada por el registro DATAX0 (byte bajo) y DATAX1 (byte alto). La salida del eje y está formada por el registro DATAY0 (byte bajo) y DATAY1 (byte alto). La salida del eje z está formada por el registro DATAZ0 (byte bajo) y DATAZ1 (byte alto). Como el sensor está configurado con una resolución de 10 bits sólo se utilizan los dos bits menos significativos del byte alto, que serán los dos bits más significativos de la palabra final, puesto que

se concatenan con el byte bajo entero. La salida tiene un formato con signo en complemento a 2. Por ello los datos se reconstruyen y almacenan en las variables eje_x, eje_y, eje_z, que son signed int16. En la figura 5.14 se pueden ver los formatos de datos disponibles en el ADXL345.

```
//Lectura Ejes
void lectura_ejes ( void )
{
readADX ( DATAX0 , 6 , datos );
//lee los registros de los ejes x,y,z byte a byte
eje_x =(( signed int16 ) datos [ 1 ])<< 8 ;
eje_x = eje_x |(( signed int16 ) datos [ 0 ]);
eje_y =(( signed int16 ) datos [ 3 ])<< 8 ;
eje_y = eje_y |(( signed int16 ) datos [ 2 ]);
eje_z =(( signed int16 ) datos [ 5 ])<< 8 ;
eje_z = eje_z |(( signed int16 ) datos [ 4 ]);
//reconstruccion de los bytes para formar la palabra
        de 10 bits de cada eje
}
```

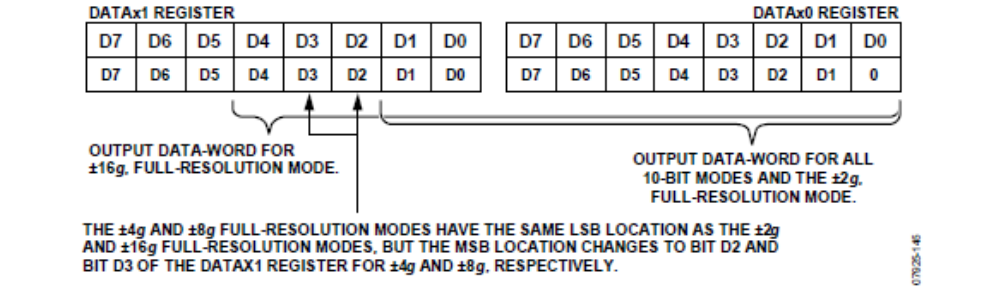


Figure 48. Data Formatting of Full-Resolution and ±2 g, 10-Bit Modes of Operation When Output Data Is Right Justified

Figura 5.14: Formato de datos del ADXL345

Función que toma 10 muestras de la posición inicial, ejes x, y, z, y hace la media aritmética de ellas para tener un valor inicial promedio de cada eje.

```
//Lectura Inicial
void lectura_inic ( void )
{
int i , j , k , l , m ;
for ( j = 0 ; j < 10 ; j ++) //inicializacion array x
{ lect_x [ j ]= 0 ;}
for ( k = 0 ; k < 10 ; k ++) //inicializacion array y
{ lect_y [ k ]= 0 ;}
for ( l = 0 ; l < 10 ; l ++) //inicializacion array z
```

```
{ lect_z [ l ]= 0 ;}
for ( i = 0 ; i < 10 ; i ++) //toma 10 muestras
        de la posicion inicial
{
delay_ms ( user_delay );
lectura_ejes ();
lect_x [ i ]= eje_x ;
lect_y [ i ]= eje_y ;
lect_z [ i ]= eje_z ;
}
for ( m = 0 ; m < 10 ; m ++) //media aritmetica
        de cada eje
{
suma_x += lect_x [ m ];
suma_y += lect_y [ m ];
suma_z += lect_z [ m ];
}
media_inic_x =( suma_x / 10 );
media_inic_y =( suma_y / 10 );
media_inic_z =( suma_z / 10 );
}
```

Función que ajusta escribiendo los registros de Offset (x, y, z) del ADXL según fórmula del datasheet. Cada uno de estos registros tiene un formato de byte con signo, y ese valor es sumado algebraicamente de forma automática a la salida de datos de los ejes (DATAX0, DATAX1, etc). Los ejes x e y son ajustados usando el valor promedio inicial de cada eje. El eje z, al estar sometido a la fuerza de la gravedad en la posición inicial estándar, hace que su valor promedio inicial sea a 1g, por lo que hay que calcular el valor a 0g antes de ajustar el offset.

```
//Ajuste Offsets
void ajuste_offs ( void ){
ajuste_x =( signed int )(-( floor ( media_inic_x / 4 )));
//ajuste de offset segun datasheet ,
        Xoff = -Round(medida_x_cerog/4)
writeADX ( OFSX , ajuste_x );
ajuste_y =( signed int )(-( floor ( media_inic_y / 4 )));
writeADX ( OFSY , ajuste_y );
z_cerog = media_inic_z -( signed int16 ) sensitivity_z ;
ajuste_z =( signed int )(-( floor ( z_cerog / 4 )));
writeADX ( OFSZ , ajuste_z );
//segun datasheet , z_cerog=medida_z_unog - sensitivity_z
//Zoff = -Round(z_cerog/4)
//pues en posicion de reposo gravedad hace eje z 1g
}
```

Función que realiza la lectura actual de los datos de los tres ejes, aunque sólo se va a trabajar con los ejes x e y para el movimiento, puesto que el eje z se reserva para reconocer los cambios bruscos de aceleración en ese eje (tap o click) mediante interrupción. Para interpretar correctamente el movimiento del dispositivo según lo expuesto en el planteamiento se trabaja con el valor de la salida de cada eje (x,y), primero para determinar el cuadrante en que se encuentra el dispositivo. Ver figura 5.15. Seguidamente se toman valores absolutos para trabajar más cómodamente y se evalúan las siguientes condiciones. Si el valor de los ejes x e y en valor absoluto es menor que el umbral de reposo (r), se está en la Zona de Reposo, por lo que no habrá movimiento. Por ahora para visualizar las acciones se crea la variable "accion" de tipo string donde se almacenará cada acción en una cadena de caracteres mediante la función strcpy(). En este caso, la acción es REPOSO. Si no se da la condición anterior, se evalúa si el valor de los ejes x e y en valor absoluto es menor que el umbral de velocidad (v). En caso afirmativo quiere decir que se está en la Zona de Movimiento Normal. Según el cuadrante y según el espacio entre ejes en el que se encuentre el dispositivo, se realizarán las acciones UP, UP_RIGHT, RIGHT, DOWN, etc. Si tampoco se da la condición anterior, entonces se está en la Zona de Movimiento Rápido. Según el cuadrante y según el espacio entre ejes en el que se encuentre el dispositivo, se realizarán las acciones UPfast, UP_RIGHTfast, RIGHTfast, DOWNfast, etc.

```
//Lectura Actual
void lectura_actual ( void )
{
lectura_ejes ();
eje_x_real =(- eje_x );
//se cambia el signo al eje_x para corresponder
        con la realidad fisica
if ( eje_x_real >= 0 ) //determinacion de cuadrante
{
if ( eje_y >= 0 )
cuadrante = 1 ;
else
cuadrante = 4 ;
}
else
{
if ( eje_y >= 0 )
cuadrante = 2 ;
else
cuadrante = 3 ;
}
eje_x_abs = abs ( eje_x );
```

```
eje_y_abs = abs ( eje_y );
//se toma el valor absoluto para trabajar mas facilmente
        en cada cuadrante
if ( eje_x_abs < umbral_reposo && eje_y_abs <
        umbral_reposo )
//condicion reposo
{
strcpy ( accion , "REPOSO_______" );
}
else if ( eje_x_abs < umbral_velocidad && eje_y_abs <
        umbral_velocidad )
//condicion para movimiento normal eje x , eje y
{
switch ( cuadrante )
{
case 1 :
{
if ( eje_y_abs >( 2 * eje_x_abs ))
{
strcpy ( accion , "UP___________" );
}
else if ( eje_y_abs >( 0 . 5 * eje_x_abs ))
{
strcpy ( accion , "UP_RIGHT_____" );
}
else
{
strcpy ( accion , "RIGHT________" );
}
break ;
}
case 2 :
{
if ( eje_y_abs >( 2 * eje_x_abs ))
{
strcpy ( accion , "UP___________" );
}
else if ( eje_y_abs >( 0 . 5 * eje_x_abs ))
{
strcpy ( accion , "UP_LEFT______" );
}
else
{
strcpy ( accion , "LEFT_________" );
```

```
}
break ;
}
case 3 :
{
if ( eje_y_abs >( 2 * eje_x_abs ))
{
strcpy ( accion , "DOWN_________" );
}
else if ( eje_y_abs >( 0 . 5 * eje_x_abs ))
{
strcpy ( accion , "DOWN_LEFT____" );
}
else
{
strcpy ( accion , "LEFT_________" );
}
break ;
}
case 4 :
{
if ( eje_y_abs >( 2 * eje_x_abs ))
{
strcpy ( accion , "DOWN_________" );
}
else if ( eje_y_abs >( 0 . 5 * eje_x_abs ))
{
strcpy ( accion , "DOWN_RIGHT___" );
}
else
{
strcpy ( accion , "RIGHT________" );
}
break ;
}
}
}
else //movimiento rapido eje x , eje y
{
switch ( cuadrante )
{
case 1 :
{
if ( eje_y_abs >( 2 * eje_x_abs ))
```

```
{
strcpy ( accion , "UPfast_______" );
}
else if ( eje_y_abs >( 0 . 5 * eje_x_abs ))
{
strcpy ( accion , "UP_RIGHTfast__" );
}
else
{
strcpy ( accion , "RIGHTfast____" );
}
break ;
}
case 2 :
{
if ( eje_y_abs >( 2 * eje_x_abs ))
{
strcpy ( accion , "UPfast_______" );
}
else if ( eje_y_abs >( 0 . 5 * eje_x_abs ))
{
strcpy ( accion , "UP_LEFTfast___" );
}
else
{
strcpy ( accion , "LEFTfast_____" );
}
break ;
}
case 3 :
{
if ( eje_y_abs >( 2 * eje_x_abs ))
{
strcpy ( accion , "DOWNfast_____" );
}
else if ( eje_y_abs >( 0 . 5 * eje_x_abs ))
{
strcpy ( accion , "DOWN_LEFTfast_" );
}
else
{
strcpy ( accion , "LEFTfast_____" );
}
break ;
```

```
}
case 4 :
{
if ( eje_y_abs >( 2 * eje_x_abs ))
{
strcpy ( accion , "DOWNfast_____" );
}
else if ( eje_y_abs >( 0 . 5 * eje_x_abs ))
{
strcpy ( accion , "DOWN_RIGHTfast" );
}
else
{
strcpy ( accion , "RIGHTfast_____" );
}
break ;
}
}
}
}
```

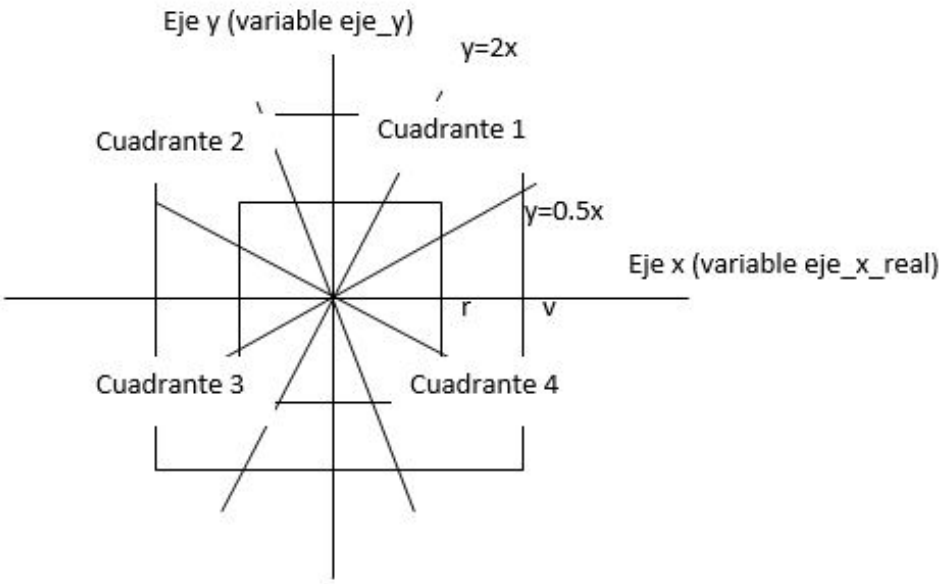


Figura 5.15: Cuadrantes en las zonas de funcionamiento

En la función main viene reflejada la secuencia de acciones que se ejecutan. Después de configurar el PIC y el ADXL hay una lectura de los registros principales del sensor que sirve sólo de comprobación de que se ha configurado bien. Se efectúa la lectura inicial de los ejes y se ajustan los offsets. Seguidamente se

entra en un bucle infinito en el que se efectúa de forma continuada la lectura de los ejes en cada momento y su correspondiente interpretación y envío de acciones a ejecutar, en este caso sólo es una escritura de caracteres en la variable "accion".

```
/* Programa Principal */
void main ()
{
int j ;
for ( j = 0 ; j < 6 ; j ++) //inicializacion array datos
{ datos [ j ]= 0 ;}
config_PIC (); //configuracion PIC
config_ADX (); //configuracion ADXL
readADX ( DEVID , 1 , datos ); //lectura registros
        principales adx
jdevid = datos [ 0 ];
readADX ( BW_RATE , 2 , datos );
jbw_rate = datos [ 0 ];
jpower_ctl = datos [ 1 ];
readADX ( DATA_FORMAT , 1 , datos );
jdata_format = datos [ 0 ];
lectura_inic (); //lectura inicial de ejes
ajuste_offs (); //ajuste de offsets
flag = 1 ;
while ( TRUE )
{
delay_ms ( user_delay );
lectura_actual (); //lectura actual,interpretacion datos
flag = 2 ;
}
}
```

Programa adx_pic_tap.c

Objetivo:

Detectar los cambios bruscos de aceleración en el eje z, llamados "tap", que se generen como interrupción en el ADXL. Además se leen los ejes x e y de igual forma que en el programa adx_pic.c, para interpretar el movimiento del dispositivo y realizar la acción correspondiente según la zona en la que se encuentre.

Planteamiento:

Interrupción Tap; figura 5.16. El ADXL tiene como característica que es capaz de detectar cambios bruscos de aceleración en cualquiera de los tres ejes (tap ó single tap). Se puede configurar esa detección ajustando un valor umbral de tap (registro THRESH_TAP) y la duración del tap (registro DUR). Si se detecta una aceleración mayor que el valor umbral en un tiempo menor que en el especificado en la duración, el ADXL manda una interrupción poniendo a nivel alto uno de los pines INT. También detecta double taps, pero esto se deja por el momento. En nuestro caso se va a utilizar sólo el eje z para detección de taps, puesto que el eje x e y son utilizados para interpretar el movimiento del dispositivo.

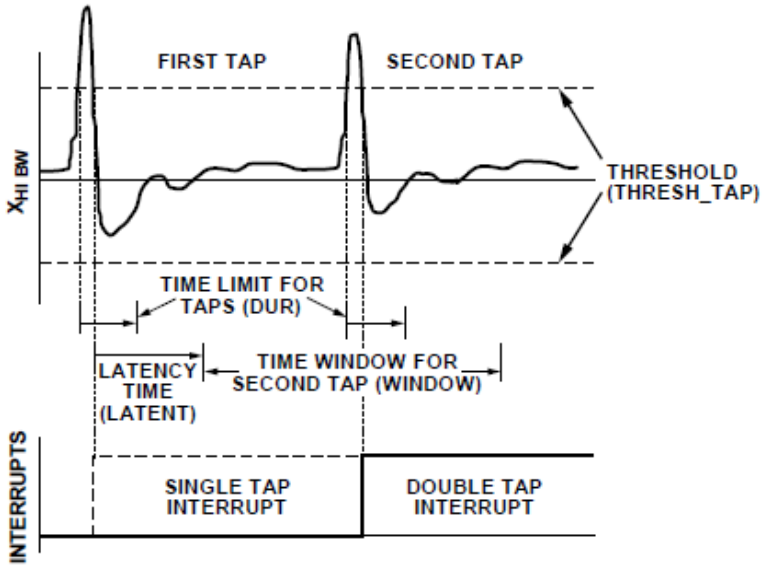


Figura 5.16: Interrupciones Tap y doble Tap en el ADXL345

Modificación de Registros:

Se escribe el registro TAP_AXES con valor 0x01 para indicar que sólo se habilita el tap del eje z.

El registro INT_MAP indica a qué pin de salida se manda cada interrupción. Single_Tap se manda al pin INT1 y Double_Tap al pin INT2. El resto de interrupciones no se van a utilizar, así que es indiferente a qué pin se manda cada una. Valor 0x20.

El registro INT_ENABLE habilita las interrupciones. Con el valor 0x40 sólo se habilita el Single_Tap. Se activará en el momento adecuado del código, no en la configuración inicial.

El registro ACT_TAP_STATUS es de sólo lectura, el bit 0 indica que la fuente del tap es el eje z. Si hay interrupción en eje z se pone a 1 ese bit. Debe

ser leído antes de borrar la interrupción.

El registro INT_SOURCE es de sólo lectura, el bit 6 indica si se ha producido o no un Single_Tap. Al ser leído se borra la interrupción.

Al registro TRIS_A del PIC se le da el valor 0xFF para asegurar que todos los pines del puerto A actúan como entradas, puesto que el PIN_A0 va a recibir la señal digital del pin INT1 del sensor, y el PIN_A1 va a recibir la señal digital del pin INT2 del sensor.

En la figura 5.17 se puede ver el diagrama de flujo de este programa.

Modificación del código:

El código es básicamente el mismo que en el programa adx_pic.c, al que se le añade la función Single Tap.

Esta función controla si se entra en ella con la variable flag_tap, cuenta las veces que se efectúa un tap con la variable numtaps, lee los registros ACT_TAP_STATUS e INT_SOURCE del sensor, y realiza la acción TAP. En este caso, sólo se mandan dichos caracteres a la variable accion.

```
//Single Tap
void accion_tap ( void )
{
flag_tap = 1 ;
numtaps += 1 ;
readADX ( ACT_TAP_STATUS , 1 , datos ); //bit0=1 eje z
        fuente del tap
jacttap_status = datos [ 0 ];
readADX ( INT_SOURCE , 1 , datos ); //bit6=1 single tap;
        borra la interrupcion
jint_source = datos [ 0 ];
strcpy ( accion , "TAP----------" );
}
```

Se puede observar en la función main del código cómo se habilita la interrupción después de las configuraciones, la lectura inicial y el ajuste de offsets, escribiendo en el registro INT_ENABLE. Luego hay un bucle infinito en el que se controla con la función input(pin) si hay tap. En caso afirmativo se ejecuta la función accion_tap(). En caso negativo se realiza la lectura actual de los ejes x e y, interpretando su movimiento con las acciones correspondientes.

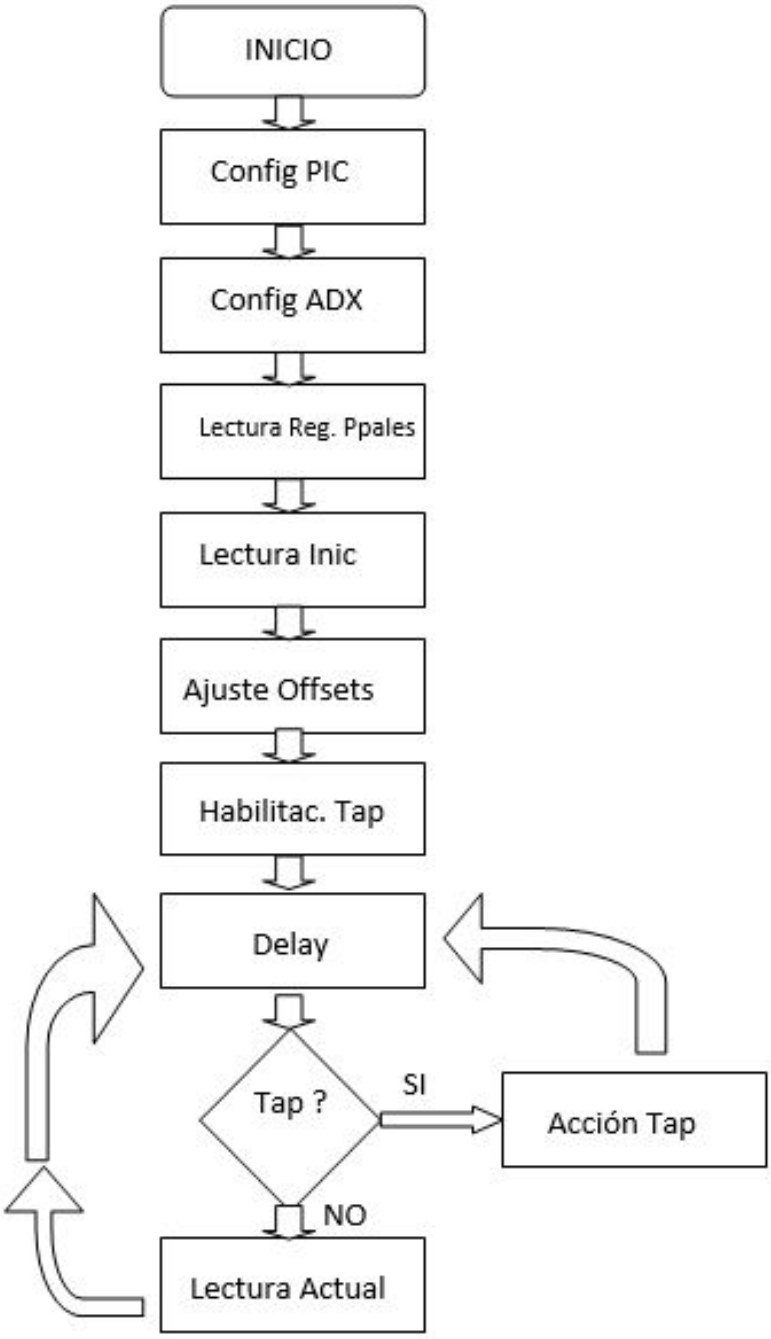


Figura 5.17: Diagrama de flujo del programa adx_pic_tap.c

5.1.3. Programación microcontrolador–módulo Bluetooth

Programa: pic_bt.c

Objetivo: Se programa en el PIC un programa sencillo para enviar varios printf() a través del módulo Bluetooth.

Planteamiento:

Se configura adecuadamente el PIC con 'set_tris' para establecer los pines de los puertos como entradas o salidas. A continuación se escriben dos bloques de código, en el primero se envían números en hexadecimal y en el segundo se envían las cadenas de caracteres UP y DOWN. La idea es detectar estos datos en el ordenador, por Bluetooth, y visualizarlos en pantalla en un programa terminal de recepción de datos serie, tipo Hyperterminal de Windows.

En pic_bt.h se configura la UART del PIC:

```
#use rs232(baud=9600, bits=8, stop=1, parity=N,
        xmit=PIN\_C6, rcv=PIN\_C7)
```

También, muy importante, se establece la directiva NODEBUG, puesto que a partir de ahora no se va a depurar con el programa MPLAB IDE y el PIC, sino que se harán pruebas directas con el PIC, el módulo Bluetooth y Linux. Por tanto es necesario escribir el programa en el PIC con el menú Programmer–Program del MPLAB IDE y no con el menú Debug como hasta ahora.

Para probar este programa se van a utilizar los sistemas operativos Windows y Linux (distribución Ubuntu), ya que de momento sólo se trata de detectar datos en serie enviados por Bluetooth directamente. Para ello se van a utilizar los programas Docklight y Moserial, para Windows y Linux respectivamente (ver figuras 5.18 y 5.19), que son terminales RS232 que muestran en pantalla cualquier dato que se reciba en el ordenador mediante una conexión serie. Estas pruebas se describen en el capítulo 7.

Wiimote™/ Cwiid

El planteamiento inicial a la hora de escribir un programa que transfiriera los datos de movimiento del dispositivo al ordenador, era simular con nuestro dispositivo el funcionamiento del mando inalámbrico Wiimote™ de Nintendo™ para interaccionar con el programa Cwiid, el cual estaría instalado en el ordenador. Cwiid es un paquete open-source de Linux que recibe e interpreta los movimien-

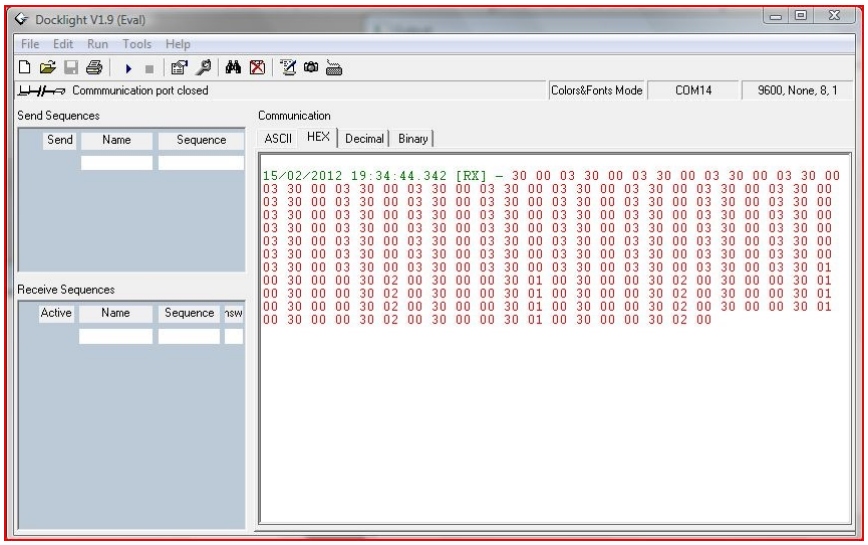


Figura 5.18: Pantalla del programa Docklight

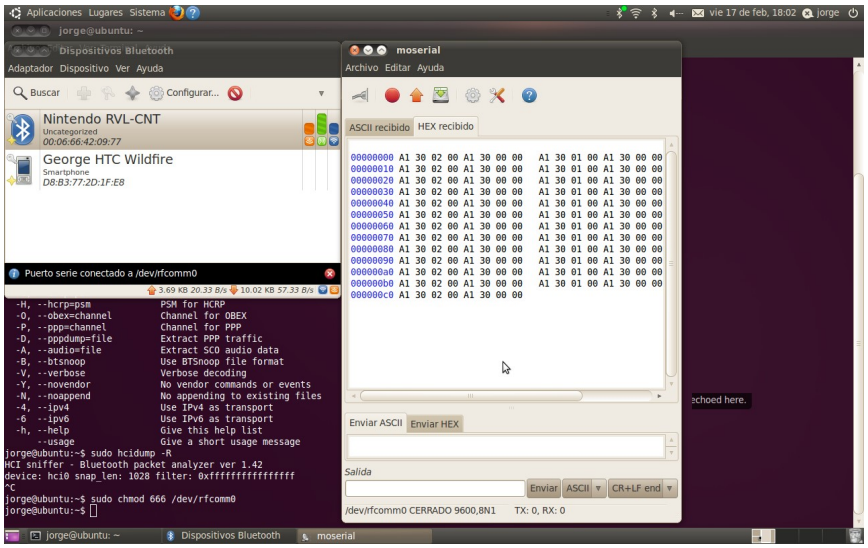


Figura 5.19: Pantalla del programa Moserial

tos y pulsaciones del Wiimote™.

Para ello se han invertido muchos días de búsqueda, investigación y estudio sobre el funcionamiento del Wiimote™ y sobre el funcionamiento de Cwiid. Cwiid está compuesto de los programas lswm, wminput y wmgui, incluyendo para su funcionamiento decenas de códigos fuente con miles de líneas de código, lo que hizo difícil su comprensión. Aún así, se modificó el código de Cwiid en ciertas líneas para encajar con las características del módulo Bluetooth RN42, y se probaron otras modificaciones de código relacionado con sockets. Finalmente quedó todo en un punto muerto puesto que la comunicación implementada en Cwiid es a través del protocolo L2CAP de Bluetooth, mientras que el RN42 aunque posee ese protocolo, envía los datos a través del protocolo RFCOMM de Bluetooth.

A continuación se muestran algunas de las herramientas Linux utilizadas para la investigación Bluetooth–RN42–Wiimote™.

- hidd –search : Busca dispositivos Bluetooth.
- hcitool scan : Busca dispositivos Bluetooth. Figura 5.20.
- sudo hcitool info <address> : Muestra información de los servicios Bluetooth. Figura 5.20.
- sudo hcidump –X –V : Lista información de paquetes Bluetooth recibidos. Figuras 5.21 y 5.22.
- sdptool browse –l2cap : Busca dispositivos Bluetooth con protocolo L2CAP y muestra características. Figuras 5.23 y 5.24.

5.2. Programación final para movimiento del cursor

El sistema operativo (S.O.) que se va a utilizar en el ordenador (PC) para comunicar con el dispositivo, es el S.O. Linux, distribución Ubuntu.

5.2.1. Programación microcontrolador

Programa de prueba: pic_bt_pointer.c

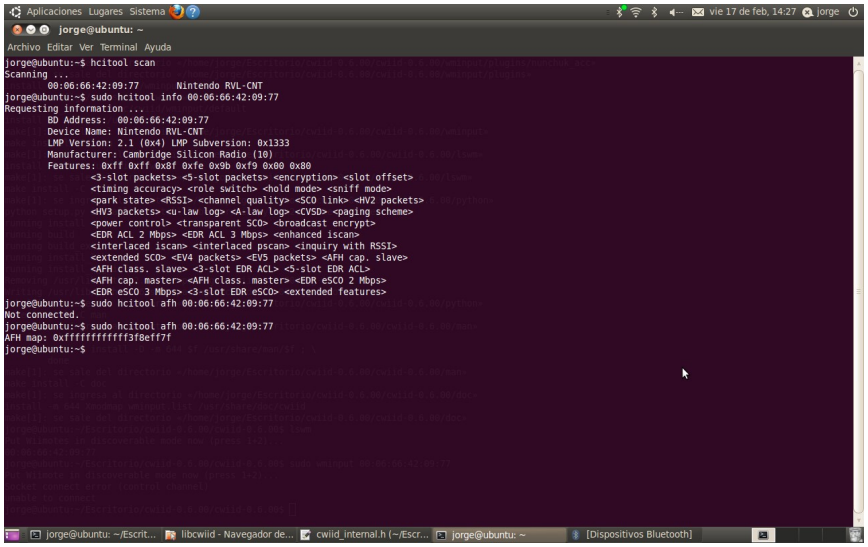


Figura 5.20: Pantalla de la herramienta Linux hcitool

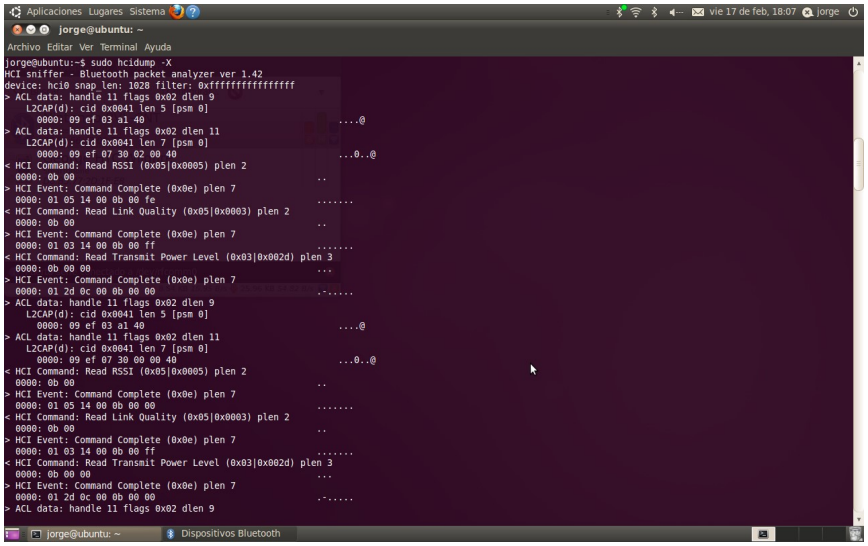


Figura 5.21: Pantalla de la herramienta Linux hcidump para el RN42

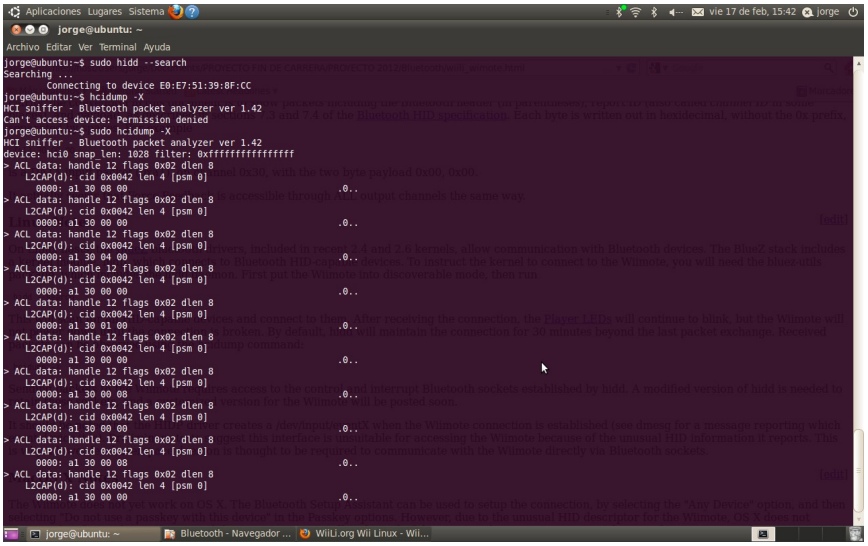


Figura 5.22: Pantalla de la herramienta Linux hcidump para el Wiimote™

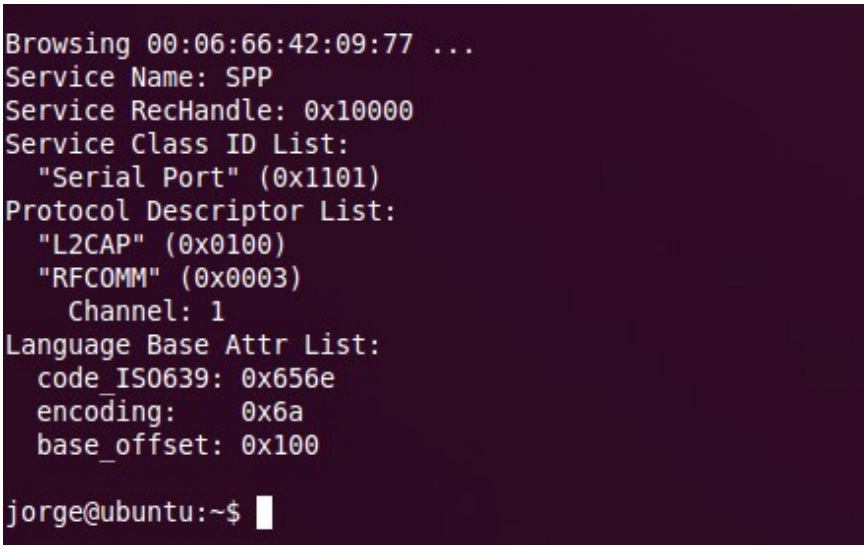


Figura 5.23: Pantalla de la herramienta Linux sdptool para el RN42

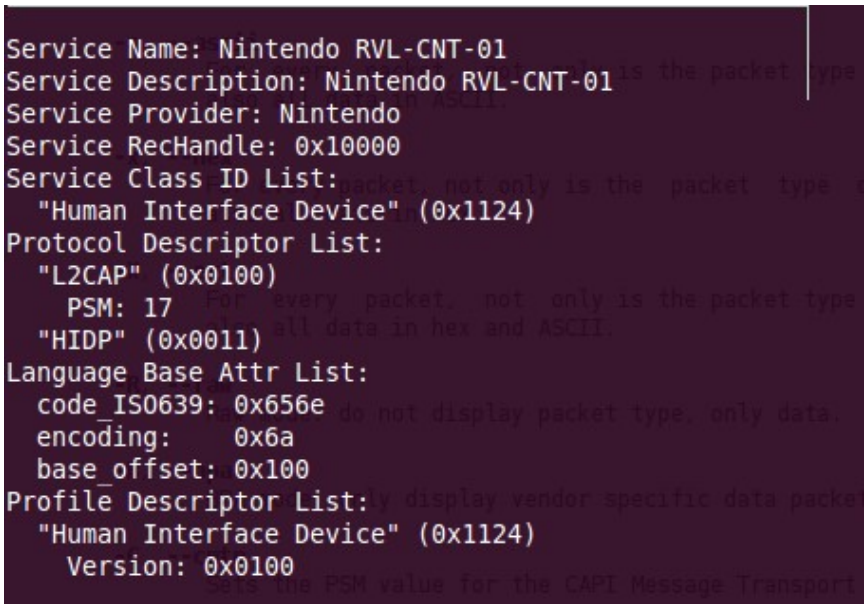


Figura 5.24: Pantalla de la herramienta Linux sdptool para el Wiimote™

Programa definitivo: adx_pic_bt.c

El objetivo de ambos programas es probar todo el hardware del dispositivo, comprobando que funciona correctamente todo el proceso de recogida, tratamiento y envío de datos desde el dispositivo. Y con la función de transformar el movimiento del dispositivo en un movimiento equivalente del cursor del ordenador.

En cuanto al PIC se realizan dos versiones de programa. La primera versión llamada pic_bt_pointer.c hace un recorrido de prueba con el cursor efectuando un ciclo cerrado.

La versión definitiva, adx_pic_bt.c, utiliza el dispositivo completo acelerómetro–PIC–Bluetooth, para enviar los datos correspondientes al movimiento del sensor, y que esta información sea recogida por el cliente del ordenador para efectuar el movimiento correspondiente asociado con el cursor del ordenador. Como se puede comprobar, este programa se basa en su totalidad en el programa adx_pic_tap.c, siendo realmente su continuación natural. La explicación detallada del código por tanto es la misma en estos dos programas, exceptuando las siguientes modificaciones importantes. Se elimina la zona de movimiento rápido, quedando por tanto sólo la zona de reposo y la zona de movimiento normal. Se ajusta mejor el valor umbral de reposo. Se añade un printf() de un caracter asociado a cada acción de movimiento; este caracter será el que reciba el programa cliente para identificar el movimiento concreto.

Los programas cabecera son respectivamente, pic_bt_pointer.h y adx_pic_bt.h.

Recordar que ahora se establece la directiva NODEBUG puesto que no se va a depurar el programa con MPLAB IDE, sino que directamente se van a hacer pruebas con el programa cliente del PC.

5.2.2. Programación cliente del PC

Programa: rn42_pointer.c.

Se procede a crear desde cero un programa cliente en Linux para comunicarse con el módulo Bluetooth RN42 mediante sockets. Un socket, muy resumidamente, es un sistema para comunicar dos ordenadores (que tengan sistema operativo Linux) usando descriptores de ficheros estándar de Unix [45].

El programa del PC que se conecta con el módulo Bluetooth es un programa cliente, ya que el RN42 está continuamente enviando datos y es el ordenador el que tiene que decir a qué dirección se quiere conectar.

Se programa con la estructura: socket–connect–recv.

El protocolo a usar es el RFCOMM de Bluetooth, que es por el que envía datos el RN42.

Se introduce el X-programming, o programación por ventanas Linux, que usa las siguientes funciones Linux para manejar el cursor del ordenador:

- XWarpPointer() : mueve el puntero del cursor
- XOpenDisplay()
- XRootWindow()
- XselectInput()
- XFlush()
- XcloseDisplay()

En la figura 5.25 se puede ver la línea de compilación utilizada.

```
jorge@ubuntu:~/PFC_ubuntu/codigo/servidor$ gcc rn42_pointer.c -o rn42_pointer -lX11 -lbluetooth
jorge@ubuntu:~/PFC_ubuntu/codigo/servidor$ ls
rn42_client  rn42_client.c  rn42_pointer  rn42_pointer.c  rn42_server  rn42_server.c  xwarp_prueba  xwarp_prueba.c
jorge@ubuntu:~/PFC_ubuntu/codigo/servidor$
```

Figura 5.25: Línea de compilación del programa rn42_pointer.c

5.3. Programación definitiva. Valores 3 ejes

5.3.1. Programación microcontrolador

Programa: values.c

Values.c es el programa definitivo que finalmente se va a programar en el PIC. Su objetivo es realizar mediciones en cada uno de los ejes del acelerómetro en tiempo real, tratar esos datos y transmitirlos por medio del módulo Bluetooth al ordenador.

Este programa concluye el desarrollo que se ha ido trabajando con todos los programas anteriores, siendo prácticamente igual al programa adx_pic_bt.c. La diferencia está en que aquí se mandan los valores de medición de cada eje x, y, z, mediante printf(). La estructura de programa en la que se asocia una acción de movimiento según la posición del acelerómetro se mantiene por si se le diera un futuro uso en el programa cliente, sin embargo, en estas versiones definitivas, esta estructura no se va a utilizar.

Recordar que el núcleo del código se ha explicado de manera detallada en el programa adx_pic.c, que es la base de todas las versiones posteriores; capítulo 5.1.2.

5.3.2. Programación cliente del PC

Programa: bt_client.c

Como el movimiento del cursor no es la funcionalidad final se plantea realizar un programa que partiendo del anterior, rn42_pointer.c, muestre en pantalla el valor en tiempo real de cada uno de los tres ejes del acelerómetro, x, y, z.

Basado en sockets, se programa de nuevo con la estructura socket–connect–recv.

En primer lugar, se incluyen las librerías necesarias para el programa, destacando la inclusión del protocolo RFCOMM para trabajar con Bluetooth [46]. La dirección MAC del RN42 es 00:06:66:42:09:77 (ver capítulo 7), y es sólo esta dirección la que el programa establece para buscar una conexión Bluetooth. Una vez se ha conectado satisfactoriamente el programa con el RN42, se procede a la lectura de los datos enviados desde el dispositivo. Se muestran en pantalla de la siguiente manera:

[OK] (eje_x eje_y eje_z)

Así, el programa muestra los valores de inclinación de los 3 ejes de manera continua, hasta que se cierre la conexión apagando el dispositivo. Recordar de nuevo que en un futuro desarrollo estos valores se pasan directamente al software del robot para que éste efectúe los movimientos asociados correspondientes.

En la figura 5.26 se puede ver la línea de compilación utilizada.

```
jorge@ubuntu:~/PFC_ubuntu/codigo/servidor$ gcc bt_client.c -o bt_client -lbluetooth
```

Figura 5.26: Línea de compilación del programa bt_client.c

Capítulo 6

Diseño Mecánico

Una vez diseñado y fabricado el hardware que conforma la placa de circuito impreso (PCB) del dispositivo con todos los componentes montados, y una vez diseñado y probado el software que le da funcionalidad, se pasa a la fase de diseño de un soporte mecánico que contenga y proteja toda la electrónica.

El dispositivo de la interfaz hombre–máquina va a ser utilizado por una persona, de la cual se van a medir los movimientos espaciales para enviárselos al ordenador. No sería adecuado que la persona cogiera la PCB con la mano sin más, con toda la electrónica 'al aire'. Teniendo en cuenta además que la creación de la interfaz está orientada al uso por parte de personas con discapacidad o personas mayores, es imperativo que la PCB se apoye en un soporte que a su vez se acople a la persona de forma solidaria para seguir los movimientos de forma precisa.

Por ello se diseña una carcasa que simultáneamente, sirve de apoyo a la PCB, protege los componentes electrónicos dejando el hueco necesario para los componentes que interactúan con el exterior y posee un sistema para acoplarse a la persona de forma sencilla.

La idea inicial es basarse en un reloj de pulsera, es decir, crear un dispositivo a modo de reloj que se pueda acoplar a la muñeca del brazo, de tal manera que al mover ese brazo el acelerómetro de la PCB detecte las diferencias de inclinación en cada uno de los ejes x, y, z. Como además el dispositivo se alimenta con pilas botón, la apariencia del conjunto no es muy diferente al de un reloj de pulsera, y por tanto la usabilidad y facilidad de manejo para la persona será muy parecida. En el caso de que la aplicación se dirija a personas con una discapacidad en la cual no se tenga una movilidad suficiente en los brazos, el dispositivo se podría adaptar cambiando el sistema de acople a otro acorde a la parte móvil del cuerpo de esas personas. Esto sería parte de futuros desarrollos que se explican en el

capítulo 8.

Por tanto, para poder acoplar el dispositivo a modo de reloj de pulsera, se diseña la carcasa con unos apéndices con ranuras para poder pasar una correa a través de ellos.

Es importante en el diseño tener en cuenta que para sujetar la PCB a la carcasa hay que utilizar algún sistema mecánico sencillo, en este caso tornillos. Esto ya se tuvo en cuenta para el diseño de la PCB, por lo que se tienen 4 agujeros en posiciones concretas. Estos agujeros se tienen que hacer corresponder en el diseño mecánico de la carcasa, para que los tornillos puedan estar en su sitio y sujetar bien la PCB.

La decisión de llevar a cabo este diseño mecánico en el proyecto se hizo debido a que en la Universidad se dispone de una impresora 3D en la que ya se habían probado algunos diseños de objetos, y se ha pensado que se podría fabricar de manera sencilla un soporte para la PCB con un diseño propio. La impresora utilizada es una Thing-O-Matic de Makerbot de nombre MADRE (figura 6.1), y el programa de impresión asociado es el ReplicatorG. El material que utiliza la impresora 3D es el plástico ABS (Acrilonitrilo Butadieno Estireno), que es económico, versátil y del que se pueden elegir muchos colores. El resultado una vez imprimido el diseño es un plástico ABS endurecido, robusto y ligero que efectivamente puede servir de soporte para el dispositivo.

Figura 6.1: Impresora 3D Makerbot

El diseño 3D del soporte mecánico o carcasa se realiza con el programa informático OpenSCAD. Es un programa de software libre para diseño de objetos sólidos CAD (diseño asistido por ordenador) en 3D, que se basa en la programación por comandos de forma paramétrica. El programa creado necesita ser compilado antes de poder visualizar el resultado. Los comandos e instrucciones de OpenSCAD permiten diseñar formas geométricas básicas parametrizadas, y por tanto, al introducir los parámetros necesarios para cada una de esas formas, el programa los genera y se pueden visualizar. Para conseguir formas complejas se recurre a operaciones como la unión y la intersección de las formas básicas. Como es un programa basado en los parámetros del diseño y no en el aspecto final, está especialmente indicado para diseño de piezas o diseño de ingeniería. Al código del programa escrito se le denomina script. Este script es el que el programa compila para luego renderizar el objeto en 3D. Ver figura 6.2.

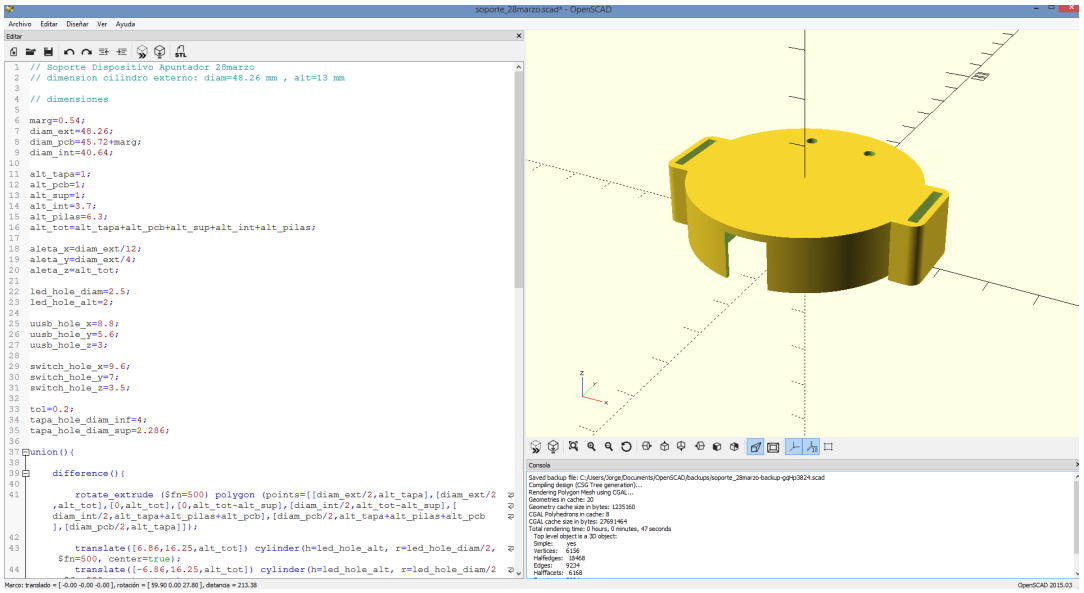


Figura 6.2: Pantalla de OpenSCAD

6.1. Diseños preliminares

Para el diseño de la pieza que va a conformar la carcasa o soporte del dispositivo es imprescindible conocer las dimensiones de la PCB con los componentes montados. Estas dimensiones se han descrito en la tabla 4.7. Lo más importante es el diámetro de la placa ya que se quiere que la PCB apoye perfectamente encajada en el soporte. Además hay que dejar el espacio necesario en altura para que quepan los componentes puesto que la carcasa será cerrada, por lo que las alturas exactas en el diseño son importantes también. Por otro lado hay que tener en cuenta que hay que dejar las aberturas necesarias para que el conector

microUSB, el interruptor deslizante y los LEDs sean accesibles. La PCB tiene 4 agujeros pensados para incorporar los tornillos que la fijan con el soporte, por tanto en el diseño hay que incorporar sus posiciones exactas.

Se han diseñado distintas versiones de la carcasa hasta que se ha llegado a la versión definitiva. En primer lugar se ha realizado un dibujo en alzado, figura 6.3, con las dimensiones principales del soporte y en el que se visualizan los espacios compartimentales donde irán alojados los componentes de la cara TOP y de la cara BOTTOM. El diseño se compone de dos piezas, el soporte donde va alojada la PCB y la tapa que cierra el conjunto. El diámetro de la PCB es de 45,72 mm, o lo que es lo mismo, 1800 mils (ver capítulo 4.3). Se establece el diámetro de la carcasa exterior en 100 mils más, y por ello la dimensión exterior del soporte es de 48,26 mm. A su vez, la PCB tiene un diámetro de plano de masa de 1600 mils, es decir, todos los componentes y pistas de rutado están dentro de ese diámetro (exceptuando el conector y el interruptor), y por tanto queda un anillo circular de 200 mils que se diseñó precisamente para establecer los agujeros de fijación y dejar espacio para el apoyo de la placa sobre la carcasa. O sea, diámetro interior: 40,64 mm.

En cuanto a las alturas, y teniendo la PCB una altura sin componentes de 1 mm, se establece esa misma dimensión para la tapa y para la parte superior del soporte. El compartimento donde van alojados los componentes de la cara TOP se diseña con 0,2 mm de margen respecto del componente más alto, que es el interruptor (3,5 mm). Igualmente, el compartimento donde van alojados los componentes de la cara BOTTOM se diseña con 0,2 mm de margen respecto del componente más alto, que es el portapilas (6,1 mm). Así, la altura total del soporte queda en un total de 13 mm.

La figura 6.4 es ya una renderización en 3D del primer diseño realizado en OpenSCAD. Como en este programa no se pueden ver cortes transversales de la pieza, ni acotaciones en el dibujo, hay que asegurarse de que la programación del código y los parámetros incluidos en él sean los correctos. Por tanto, para comprobar que las piezas tienen las dimensiones pretendidas se simula en 3D también la PCB con sus componentes montados, con sus dimensiones nominales; ver figura 6.5. De esta manera se puede superponer gráficamente la PCB con el soporte y ver si encajan correctamente.

En la figura 6.7 se puede ver el diseño de la primera versión de la tapa que cierra el soporte, y en la figura 6.6 el conjunto de soporte+tapa. En esta primera versión hay un fallo fundamental y es que la abertura para los conectores está incluida tal cual, sin tener en cuenta que al introducir la PCB en el soporte se necesita vía libre en esas zonas puesto que por diseño y para aprovechar más el espacio, el final de los conectores sobresalen un mínimo por fuera del diámetro de la tarjeta PCB. Esto se soluciona en la versión 2.

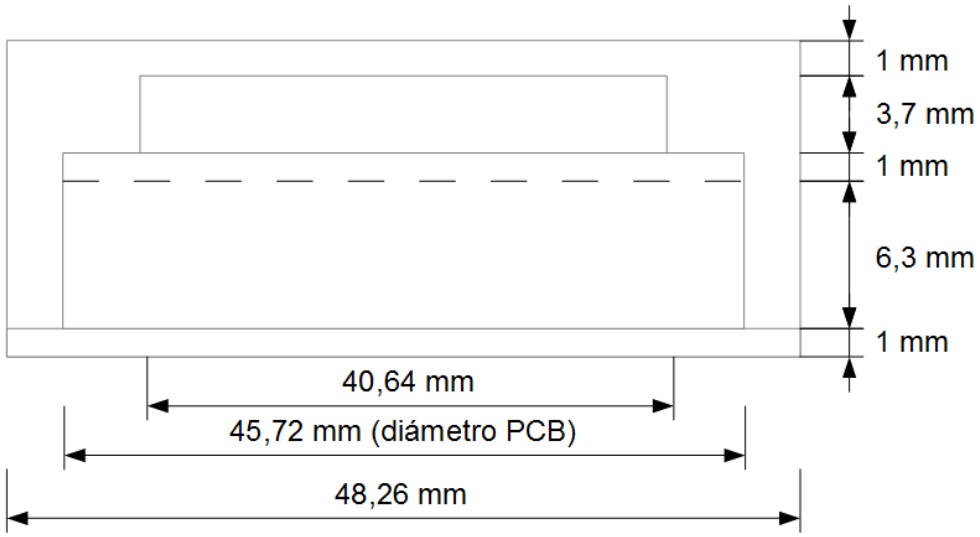


Figura 6.3: Dibujo en alzado del soporte y tapa

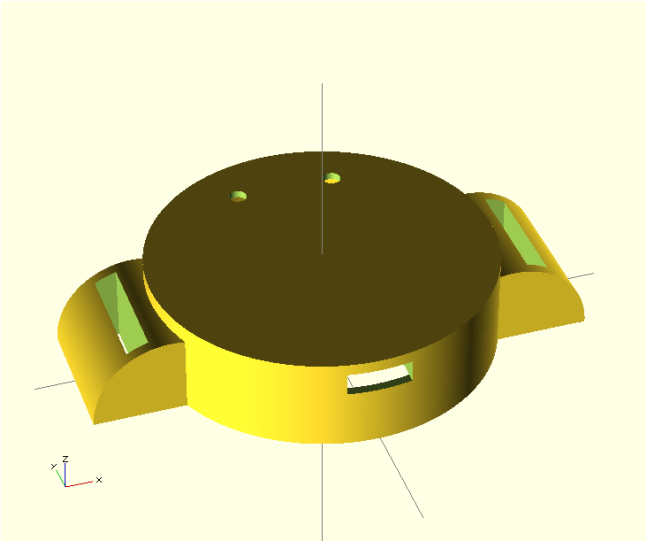


Figura 6.4: Diseño 3D del soporte. Versión 1

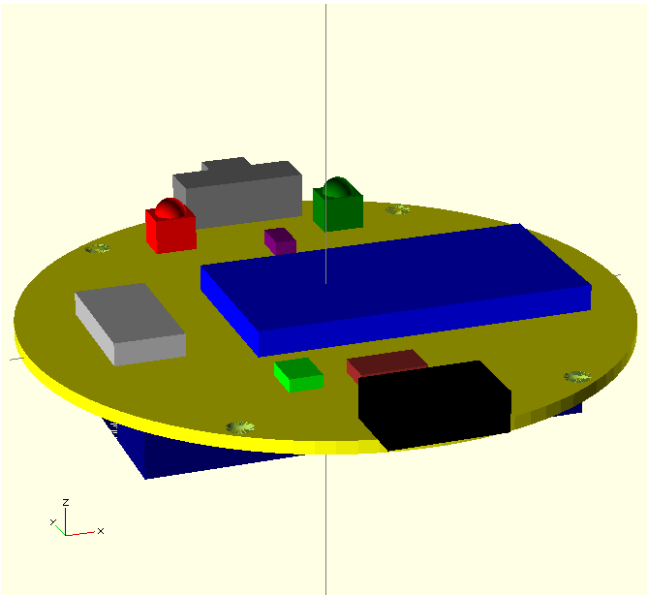


Figura 6.5: Diseño 3D de la PCB con componentes montados

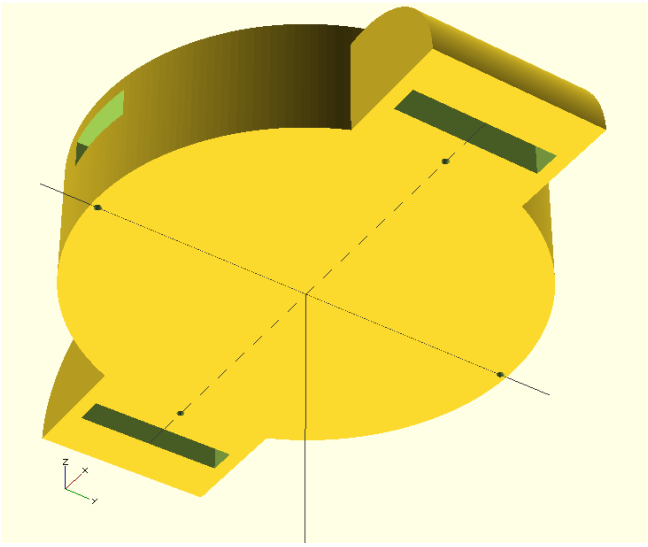


Figura 6.6: Diseño 3D del soporte+tapa. Versión 1

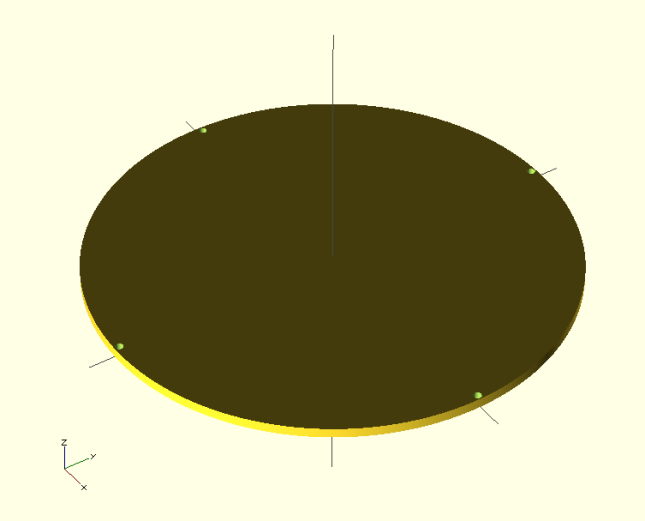


Figura 6.7: Diseño 3D de la tapa. Versión 1

La figura 6.8 representa el diseño de la versión 2 del soporte en el que las aberturas para los conectores llegan hasta la parte de abajo para solucionar el problema anterior. En la vista inferior, figura 6.9, se observa el escalón efectuado en la pieza para apoyar la PCB. Y observando esa figura junto la que representa la tapa de la versión 2 (figura 6.10) se aprecia que la solución adoptada para integrar la tapa con el soporte es que la tapa tenga el diámetro total de la carcasa y complete ésta al unirse.

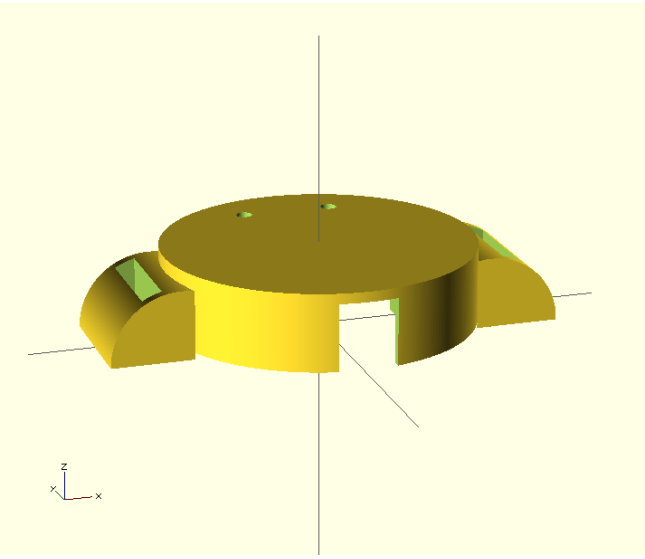


Figura 6.8: Diseño 3D del soporte. Versión 2

Hay que reseñar que los dos agujeros de la parte superior del soporte son para poder ver la luz de los LEDs a través de ellos y así saber el estado de funcionamiento del dispositivo y del módulo Bluetooth. También reseñar que

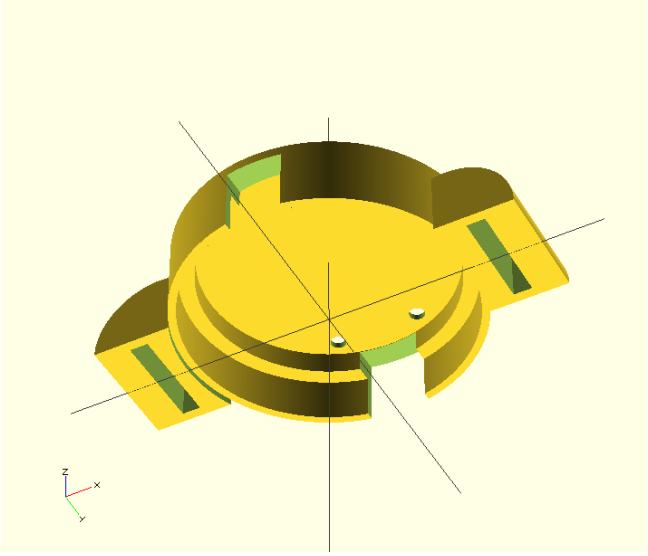


Figura 6.9: Diseño 3D del soporte. Versión 2. Vista inferior

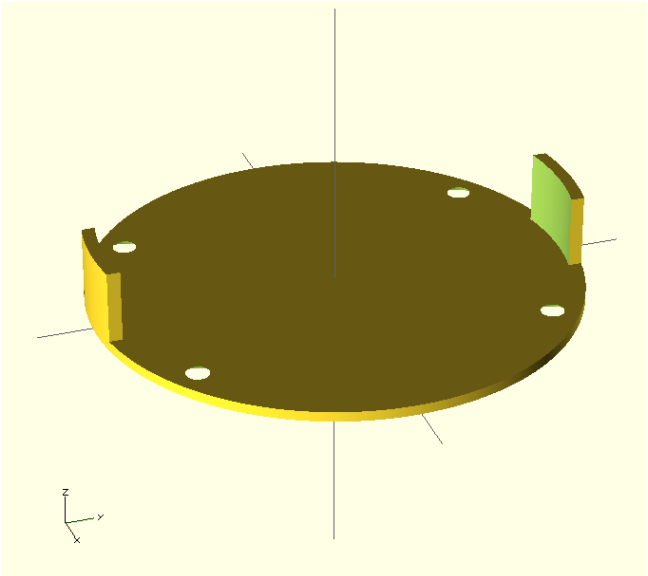


Figura 6.10: Diseño 3D de la tapa. Versión 2

en esta versión los cuatro agujeros de la tapa sí están situados en la posición exacta para que correspondan con los agujeros de la PCB, y además ya tienen la dimensión adecuada para adaptarse a los tornillos que servirán de fijación.

En este punto se decide imprimir este diseño para ver físicamente si es adecuado para insertar el dispositivo real, y para poder observar alguna posible mejora. Una recomendación importante sobre la impresora 3D por experiencias previas es conocer cómo imprime la máquina. Ésta, por lógica constructiva y 'por gravedad', imprime de abajo hacia arriba, considerando abajo la zona del diseño con más material y con más base, para poder imprimir capas nuevas que se vayan apoyando en las capas de abajo. De esta manera, en nuestro diseño el soporte lo imprime al revés según las figuras, es decir, empieza a imprimir la parte de arriba, para apoyándose en ella, ir construyendo las paredes hasta terminar en la parte de abajo de las figuras. Según este modo de operar, surge un problema puesto que los alerones o aletas laterales (que tienen las ranuras para poder pasar una correa), tienen una forma que va de menos a más al imprimirse la pieza al revés. Y la impresora no puede construir sobre el aire. Por tanto, se hace un rediseño de las aletas para solucionarlo, haciéndolas llegar hasta la base de impresión.

En las figuras 6.11, 6.12 y 6.13 se pueden ver las piezas reales una vez imprimidos los diseños con la impresora 3D. Tanto el soporte como la tapa están imprimidos en plástico ABS rojo y tienen una consistencia mecánica bastante grande. Se comprueba que, efectivamente, la PCB encaja correctamente en el soporte, y por tanto funcionalmente el diseño es correcto. Sin embargo, se observa que las aletas son demasiado grandes. Como uno de los objetivos es hacer un dispositivo lo más pequeño posible, se rediseñan las aletas para ser lo más reducidas posible. Esto se realiza ya en el diseño final.

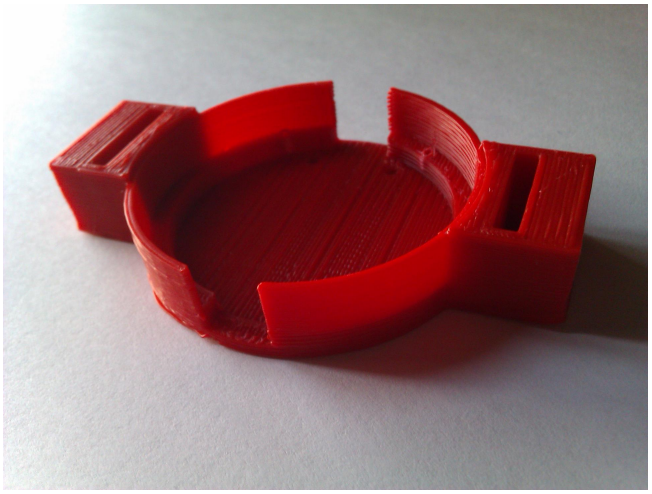


Figura 6.11: Soporte preliminar. Vista diagonal

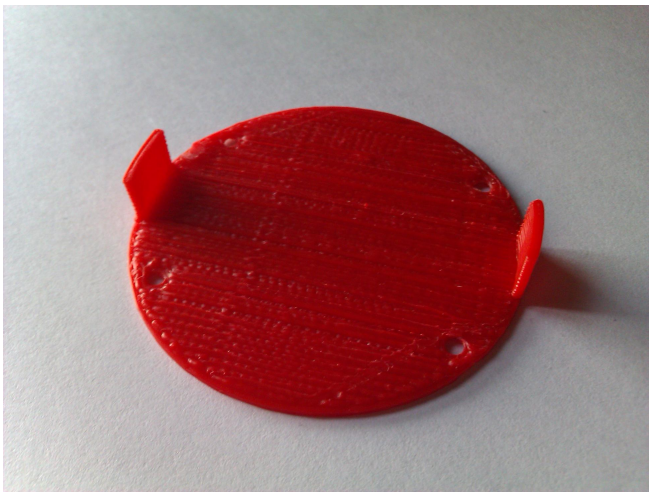


Figura 6.12: Tapa preliminar

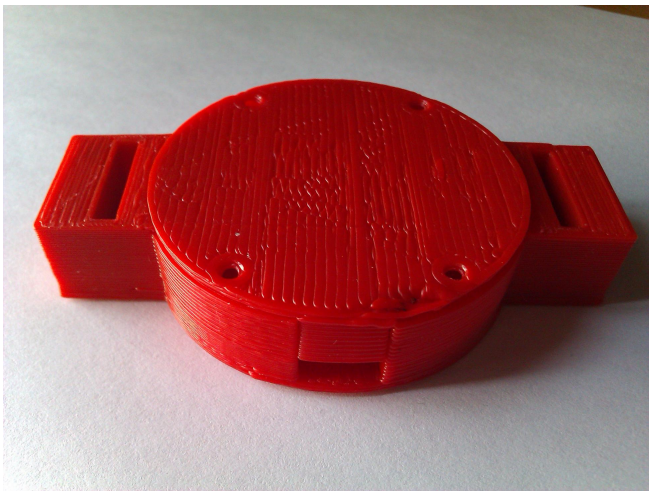


Figura 6.13: Soporte+tapa preliminar. Vista desde abajo

6.2. Diseño final

El diseño definitivo del soporte y la tapa que dan apoyo mecánico a la PCB incluye dos modificaciones importantes desde el diseño anterior.

- Los agujeros de la tapa se avellanan para facilitar la sujeción de los tornillos y se crean unos segmentos de cono para guiar a los tornillos hacia la posición en la que deben contactar con el soporte para ser atornillados. Esto se hace al comprobar físicamente que en el soporte que se imprimió anteriormente, los tornillos bailaban y no era fácil encontrar los agujeros de la PCB por los que debían pasar.
- Las aletas laterales se reducen al tamaño más pequeño posible y se redondea su forma.

6.2.1. Soporte

El diseño final del soporte se puede ver en la figura 6.14.

Las medidas finales del soporte son:

- Diámetro sin aletas: 48,26 mm.
- Altura: 13 mm.

En la figura 6.15 se puede ver el soporte definitivo real obtenido de la impresora 3D.

6.2.2. Tapa

El diseño final de la tapa se puede ver en la figura 6.16.

Las medidas finales de la tapa son:

- Diámetro: 48,26 mm.
- Altura: 1 mm.

En la figura 6.17 se puede ver la tapa definitiva real obtenida de la impresora 3D.

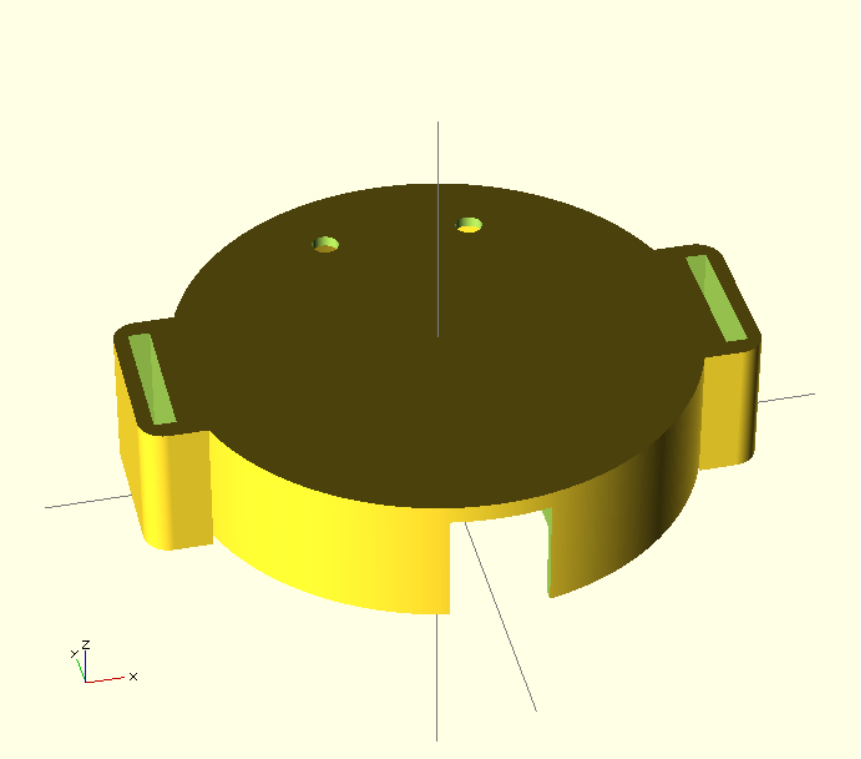


Figura 6.14: Soporte definitivo

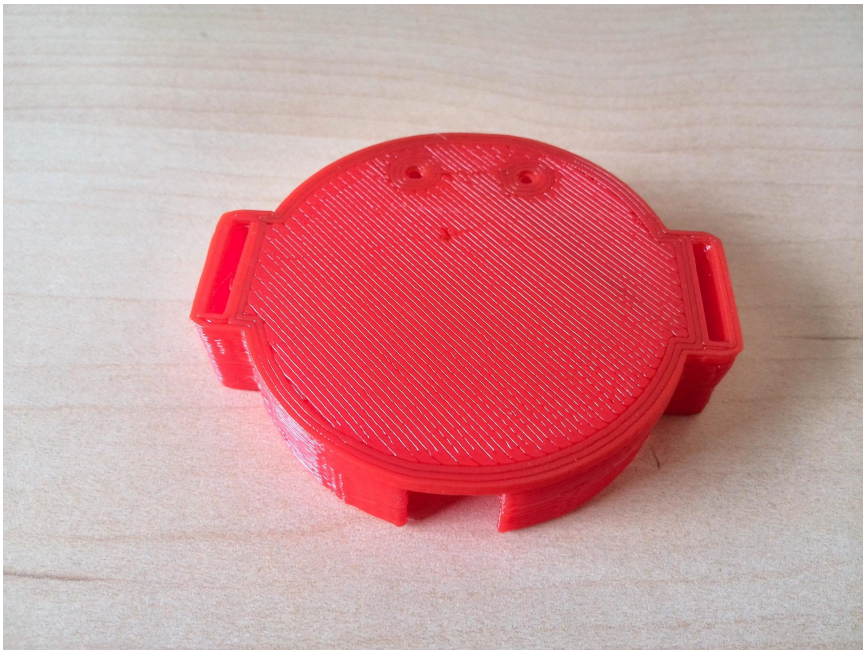


Figura 6.15: Soporte definitivo real

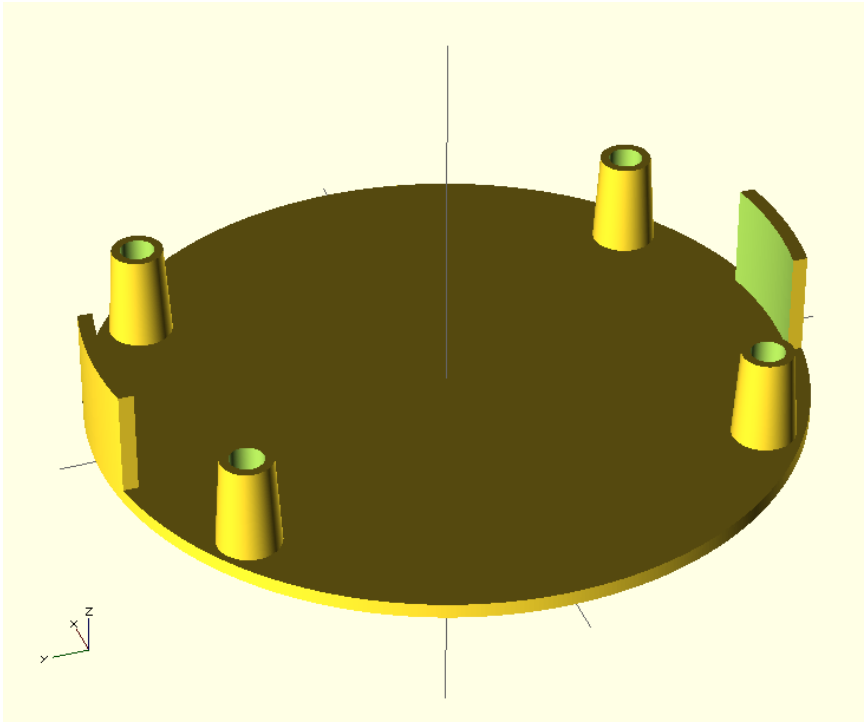


Figura 6.16: Tapa definitiva

Figura 6.17: Tapa definitiva real

Capítulo 7

Puesta en funcionamiento

7.1. Ensayos y resultados placa experimental

Una vez montada la placa experimental con todo su hardware, se ha ido probando el software de forma consecutiva. Desde la primera versión de comunicación con el acelerómetro hasta el código final se han comprobado y desarrollado los distintos códigos a raíz de las pruebas y ensayos que se muestran a continuación.

Toda la depuración del código se ha realizado con MPLAB IDE, leyendo registros y variables. La ruta de menús para programar el PIC es: Proyecto – MPLAB – CCS – Build – Connect – Program.

Programa: comunic_id.c

En la tabla 7.1 se muestran las lecturas de registros y variables realizadas para la comprobación del buen funcionamiento del código.

Resultado: Hay comunicación del PIC con el acelerómetro, y ésta es correcta.

Programa: config_reg.c

En la tabla 7.2 se muestran las lecturas de registros y variables realizadas para la comprobación del buen funcionamiento del código.

Tabla 7.1: Pruebas del código comunic_id.c. Runhalt01

Registro o Variable	Descripción
entrada: 0xE5 = 11100101	Ok !!!
TRISC: 0x93	ok
SSPSTAT: 0xC0	ok
SSPCON: 0x30	ok
SSPBUF: 0xE5	ok
INTCON: 0x01	RB6 cambia de estado

Tabla 7.2: Pruebas del código config_reg.c. Runhalt00

Registro o Variable	Descripción
TRISC: 0x93	ok
SSPSTAT: 0xC0	ok
SSPCON: 0x30	ok
INTCON: 0x00	ok
datos[0-29]	mal, no corresponden
dummy: 0xE5	ok (lectura_id)

Se realiza una modificación del código para resolver la mala correspondencia de los datos. Ver tablas 7.3 y 7.4.

Tabla 7.3: Pruebas del código config_reg.c. Runhalt01a

Registro o Variable	Descripción
TRISC: 0x93	ok
SSPSTAT: 0xC0	ok
SSPCON: 0x30	ok
INTCON: 0x01	ok
datos[0]: 0x30	ok
datos[1]: 0x00	ok
datos[2]: 0x00	ok
datos[3]: 0x00	ok
datos[4]: 0x30	ok
datos[5]: 0x18	ok
datos[6]: 0x40	ok
datos[7]: 0x00	ok
datos[8]: 0x00	ok
datos[9]: 0x00	ok
datos[10]: 0x00	ok
datos[11]: 0x00	ok
datos[12]: 0x00	ok
datos[13]: 0x01	ok
datos[14]: 0x00	ok
datos[15]: 0x0D	ok

Resultado: Funcionamiento correcto. La configuración de los registros del acelerómetro es la correcta.

Tabla 7.4: Pruebas del código config_reg.c. Runhalt01b

datos[16]: 0x00	ok
datos[17]: 0x00	ok
datos[18]: 0x60	ok
datos[19]: 0x02	ok
datos[20]: 0x00	ok
datos[21]: 0x00	ok
datos[22]: 0x00	ok
datos[23]: 0x00	ok
datos[24]: 0x00	ok
datos[25]: 0x00	ok
datos[26]: 0x00	ok
datos[27]: 0x00	ok
datos[28]: 0x00	ok

Programa: basic_measure.c

Tras un primer intento fallido se realiza una modificación del código y se hacen las pruebas de la tabla 7.5. Se muestran las lecturas de registros y variables realizadas para la comprobación del buen funcionamiento del código.

Resultado: Funcionamiento correcto.

Se activa correctamente el modo Measure del ADXL, se hacen lecturas de los registros de datos de los ejes x, y, z con la placa EN REPOSO. Se comprueba que la reconstrucción de los datos es correcta, esto es, el byte bajo y el byte alto de cada eje se combinan adecuadamente para formar la palabra de 16 bits con signo de cada eje.

Como el funcionamiento del programa es correcto, se procede a tomar medidas de los tres ejes con la placa en distintas posiciones en el espacio para comprobar que esas medidas se corresponden con el funcionamiento real del ADXL. En la figura 7.1 se ve el sentido de los movimientos de la placa experimental respecto de los ejes del acelerómetro. Estos movimientos son siempre inclinaciones o giros alrededor de los ejes x e y. Se puede observar cómo existe un problema de desalineamiento de ejes entre los ejes del acelerómetro y los ejes de la placa en el que está montado. Es una cuestión de montaje que no se podía

Tabla 7.5: Pruebas del código basic_measure.c. Runhalt01

Registro o Variable	Descripción
TRISC: 0x93	ok
SSPSTAT: 0xC0	ok
SSPCON: 0x30	ok
jdevid: 0xE5	ok
jbwrate: 0x0D	ok
jpowerctl: 0x08	ok
jdataformat: 0x00	ok
datos[0]	0000 1101 medida eje x byte bajo
datos[1]	0000 0000 medida eje x byte alto
datos[2]	0000 0110 medida eje y byte bajo
datos[3]	0000 0000 medida eje y byte alto
datos[4]	0000 0110 medida eje z byte bajo
datos[5]	0000 0001 medida eje z byte alto
eje_x	0000 0000 0000 1101(signed int16)=13 dec
eje_y	0000 0000 0000 0110(signed int16)=6 dec
eje_z	0000 0001 0000 0110(signed int16)=262 dec

evitar pero que no afecta en el funcionamiento de las pruebas a realizar puesto que todos los movimientos se van a efectuar respecto de los ejes del sensor, no de la placa.

Las pruebas de código más representativas se ven en las tablas 7.6, 7.7, 7.8, 7.9, 7.10, 7.11 y 7.12, teniendo en cuenta que además se realizan diversas pruebas seguidas moviendo la placa de manera continua.

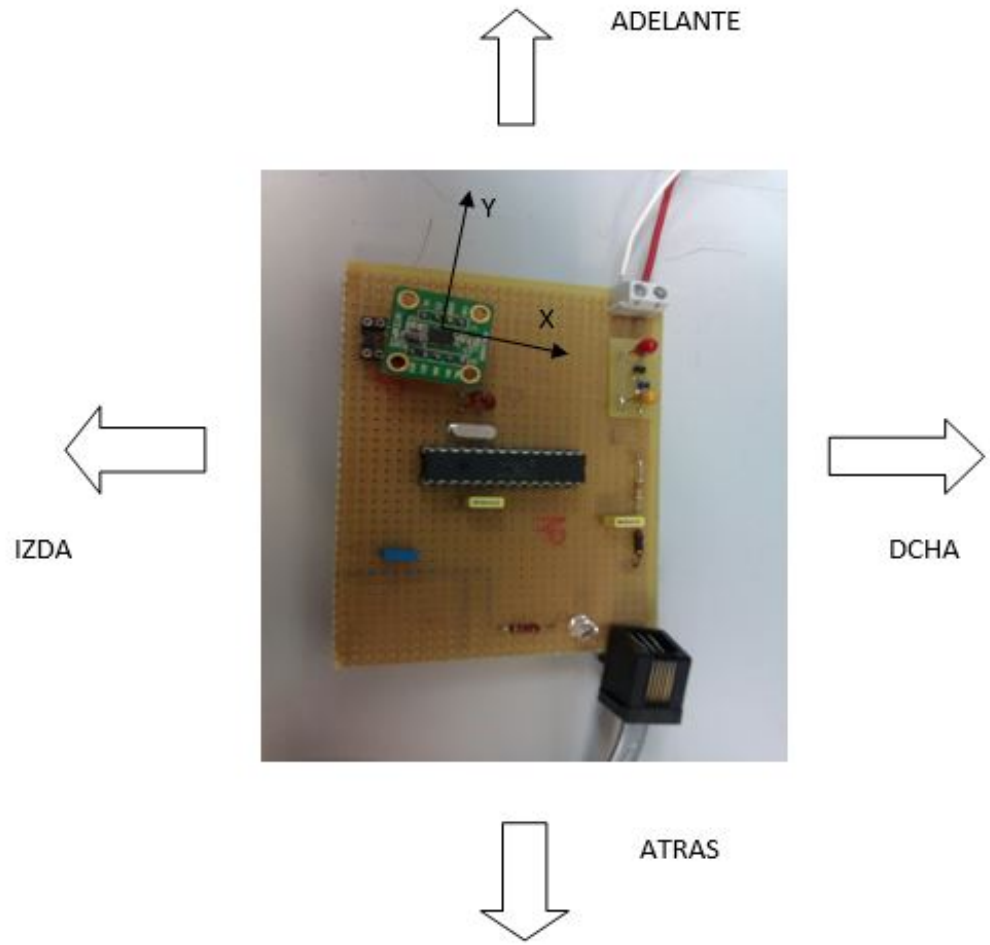


Figura 7.1: Placa experimental. Sentidos del movimiento respecto de los ejes del ADXL

Resultado: El funcionamiento es correcto. Los valores de los ejes corresponden con la posición espacial física de la placa.

Tabla 7.6: Pruebas del código basic_measure.c. Test 01
TEST 01 – REPOSO

Eje	Medida
Eje x	7 (decimal)
Eje y	2 (decimal)
Eje z	268 (decimal)

Tabla 7.7: Pruebas del código basic_measure.c. Test 05
TEST 05 – 90º ADELANTE

Eje	Medida
Eje x	-252
Eje y	-15
Eje z	-4

Tabla 7.8: Pruebas del código basic_measure.c. Test 09
TEST 09 – 90º ATRÁS

Eje	Medida
Eje x	266
Eje y	50
Eje z	31

Tabla 7.9: Pruebas del código basic_measure.c. Test 10
TEST 10 – 90º DERECHA

Eje	Medida
Eje x	-10
Eje y	260
Eje z	44

Tabla 7.10: Pruebas del código basic_measure.c. Test 11
TEST 11 – 90º IZQUIERDA

Eje	Medida
Eje x	61
Eje y	-253
Eje z	-3

Tabla 7.11: Pruebas del código basic_measure.c. Test 12
TEST 12 – BOCA ABAJO

Eje	Medida
Eje x	18
Eje y	7
Eje z	-243

Tabla 7.12: Pruebas del código basic_measure.c. Test 16
TEST 16 – 45º (+X,+Y)

Eje	Medida
Eje x	-131
Eje y	-119
Eje z	174

Programa: adx_pic.c

Las primeras pruebas se realizaron antes de incluir la zona de reposo en el código y con umbral de velocidad 128.

Primero se realiza una prueba con la placa en reposo para efectuar la comprobación de registros y variables. Tablas 7.13, 7.14, 7.15 y 7.16.

En esta primera prueba se observa que en reposo, los valores del eje x y del eje y, oscilan entre -13 y 37 en decimal, y el eje z oscila entre 249 y 298. Estas son las 10 muestras de lectura inicial. Los valores ideales deberían ser: 0 para ejes x e y; 256 para el eje z.

Se comprueba que las variables de offset tienen el valor correcto según el valor promedio inicial, se comprueba que la lectura de ejes actual se realiza correctamente, que las variables eje_x_real, eje_x_abs (5 dec), eje_y_abs (5 dec) tienen los valores correctos y que la determinación de cuadrante es correcta. Se observa que la variable 'accion' muestra la palabra Down_right, que corresponde de manera correcta con los valores actuales medidos en el eje_x_real (5) y en el eje_y (-5).

Por tanto, se observa que estando el dispositivo en reposo, el eje x y el eje y miden unos valores pequeños pero no miden 0, con los offsets ya ajustados, esto es debido a que el sensor es tan sensible y tan pequeño que como físicamente es muy difícil colocarlo de forma perfectamente horizontal va a estar siempre ligerísimamente inclinado y mide valores no nulos. Por esta razón se creó la zona de reposo.

Esta prueba se hizo antes de instaurar la zona de reposo, por ello estando la placa en reposo y al ser medidos unos valores pequeños concretos en los ejes x e y, su correspondencia es la acción de Down_right. El funcionamiento es correcto.

En las pruebas de las tablas 7.17 y 7.18, el funcionamiento es correcto. Es importante recordar que el dato eje_y mantiene los mismos signos que en una representación cartesiana habitual, mientras que el dato eje_x tiene el signo cambiado, es decir, un dato positivo indica el semiplano cartesiano izquierdo; ver figura 7.2.

A continuación, las tablas 7.19, 7.20, 7.21, 7.22, 7.23, 7.24, 7.25, 7.26, 7.27 y 7.28, muestran diversas pruebas en las que se han efectuado todas las combinaciones de movimientos, y leyendo los valores de las variables 'accion', eje_x y eje_y, se puede comprobar que el funcionamiento, efectivamente, es correcto. Se observa que el valor umbral_velocidad de 128 es pequeño, pues enseguida se salta a la zona de velocidad rápida, por lo que se ajusta posteriormente ese valor

Tabla 7.13: Pruebas del código adx_pic.c. Test 01a

TEST 01 – REPOSO (Configuración PIC y ADX)

Registro o Variable	Descripción
TRISC: 0x93	ok
SSPSTAT: 0xC0	ok
SSPCON: 0x30	ok
jbw_rate: 0x0D	ok
jdata_format: 0x00	ok
jdevid: 0xE5	ok
jpower_ctl: 0x08	ok

Tabla 7.14: Pruebas del código adx_pic.c. Test 01b

TEST 01 – REPOSO (Lectura inicial)

Registro o Variable	Descripción
lect_x	19,8,11,19,24,12,10,11,12,-13 (dec) ; ok
lect_y	21,14,37,2,12,22,7,4,-5,16 ; ok
lect_z	259,278,269,264,249,262,298,252,x,x ; ok
suma_x	113 ; ok
suma_y	130 ; ok
suma_z	2656 ; ok
media_inic_x	11 ; ok
media_inic_y	13 ; ok
media_inic_z	265 ; ok

Tabla 7.15: Pruebas del código adx_pic.c. Test 01c

TEST 01 – REPOSO (Ajuste Offsets)

Registro o Variable	Descripción
ajuste_x	-2 ; ok
ajuste_y	-3 ; ok
ajuste_z	-2 ; ok
z_cerog	2304 ; mal , pero luego ajuste_z está bien (sup. Fallo memoria MPLAB)

Tabla 7.16: Pruebas del código adx_pic.c. Test 01d

TEST 01 – REPOSO (Lectura actual)

Registro o Variable	Descripción
eje_x	-5 ; dato
eje_x_real	5 ; ok
eje_y	-5 ; dato
cuadrante	4 ; ok
eje_x_abs	5 ; ok
eje_y_abs	5 ; ok
accion	DOWN_RIGHT ; ok, corresponde con ejes_abs y cuadrante

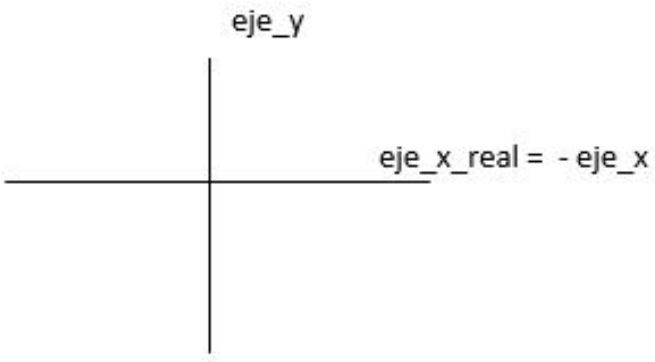


Figura 7.2: Eje x real

Tabla 7.17: Pruebas del código adx_pic.c. Test 03

TEST 03 – ADELANTE 45º (con reposo 5seg al inicio)

Registro o Variable	Descripción
Config PIC y ADX	ok
lect_x	(-5,41) ; ok (10 valores)
lect_y	(0,25) ; ok (10 valores)
Ajuste Offsets	ok
eje_x	24 ; dato
eje_y	-141 ; dato
eje_x_real	ok
eje_x_abs	ok
eje_y_abs	ok
cuadrante	3 ; ok
accion	DOWNfast ; ok, ya que el umbral velocidad es 128 y eje_y_abs>2eje_x_abs

Tabla 7.18: Pruebas del código adx_pic.c. Test 04

TEST 04 – ADELANTE 90º

Registro o Variable	Descripción
accion	DOWNfast
eje_x	39
eje_y	-236

a 200.

Tabla 7.19: Pruebas del código adx_pic.c. Test 05

TEST 05 – ATRÁS 45º

Registro o Variable	Descripción
accion	UP
eje_x	-4
eje_y	94

Posteriormente se modifica el código introduciendo la zona de reposo con un umbral de reposo de valor 30, y se realizan las pruebas de las tablas 7.29, y 7.30. Se comprueba que aquí el funcionamiento también es correcto.

Tabla 7.20: Pruebas del código adx_pic.c. Test 06

TEST 06 – ATRÁS 90º

Registro o Variable	Descripción
accion	UP fast
eje_x	9
eje_y	197

Tabla 7.21: Pruebas del código adx_pic.c. Test 07

TEST 07 – IZQUIERDA 45º

Registro o Variable	Descripción
accion	LEFTfast
eje_x	171
eje_y	-15

Tabla 7.22: Pruebas del código adx_pic.c. Test 08

TEST 08 – IZQUIERDA 90º

Registro o Variable	Descripción
accion	LEFTfast
eje_x	257
eje_y	-1

Tabla 7.23: Pruebas del código adx_pic.c. Test 09

TEST 09 – DERECHA 45º

Registro o Variable	Descripción
accion	RIGHT
eje_x	-86
eje_y	-35

Tabla 7.24: Pruebas del código adx_pic.c. Test 10

TEST 10 – DERECHA 90º

Registro o Variable	Descripción
accion	RIGHTfast
eje_x	-154
eje_y	-36

Tabla 7.25: Pruebas del código adx_pic.c. Test 11

TEST 11 – (ADELANTE+DERECHA) 45º

Registro o Variable	Descripción
accion	DOWN_RIGHT
eje_x	-125
eje_y	-63

Tabla 7.26: Pruebas del código adx_pic.c. Test 12

TEST 12 – (ADELANTE+IZQUIERDA) 45º

Registro o Variable	Descripción
accion	DOWN_LEFT
eje_x	110
eje_y	-77

Tabla 7.27: Pruebas del código adx_pic.c. Test 13

TEST 13 – (ATRÁS+IZQUIERDA) 45º

Registro o Variable	Descripción
accion	UP_LEFT
eje_x	98
eje_y	101

Tabla 7.28: Pruebas del código adx_pic.c. Test 14

TEST 14 – (ATRÁS+DERECHA) 45º

Registro o Variable	Descripción
accion	UP_RIGHT
eje_x	-57
eje_y	81

Tabla 7.29: Pruebas del código adx_pic.c. Test 16

TEST 16 – REPOSO

Registro o Variable	Descripción
accion	REPOSO
eje_x	-6
eje_y	5
Lecturas inic, sumas, medias y offsets	ok
Cuadrante	ok
Eje_x_real y absolutos	ok
Flag=2	ok

Tabla 7.30: Pruebas del código adx_pic.c. Test 19

TEST 19 – DERECHA 45º

Registro o Variable	Descripción
accion	RIGHT
eje_x	-46
eje_y	-7

Programa: adx_pic_tap.c

Con este programa se realiza primero una prueba en reposo (tabla 7.31). Después se realiza una prueba efectuando un Tap, que consiste en que, desde la posición de reposo, se acelera bruscamente la placa en la dirección del eje z. Tabla 7.32.

El hecho de que en la variable 'accion' aparezca DOWN_LEFT en lugar de TAP, es porque después de efectuarse el Tap, siguió el bucle, volvió a leer y escribió la acción correspondiente al detener la ejecución, que es "downleft" en este caso, sobreescribiendo la palabra "tap". Pero se confirma la detección del Tap por las lecturas de jacttapstatus, jintsource y flag_tap.

Tabla 7.31: Pruebas del código adx_pic_tap.c. Reposo

REPOSO

Registro o Variable	Descripción
Config (TRIS_C, jdevid, jbw_rate, etc)	ok
jacttapstatus	no ha habido Tap ; ok
jintsource	no ha habido Tap ; ok
accion	DOWN
eje_x	5
eje_y	-16 ; (sin zona de reposo implementada) ok
cuadrante	3 ; ok
lect_x inic	(2,16) ; ok
lect_y inic	(4,42) ; ok
lect_z inic	(201,264) ; ok
sumas, medias, zcerog	ok
ajuste_x	-1 ; ok
ajuste_y	-4 ; ok
ajuste_z	1 ; ok

En la siguiente prueba se efectúan varios movimientos de inclinación desde el reposo, y seguidamente se acelera bruscamente una vez la placa en la dirección del eje z, y pasados 3 segundos se vuelve a hacer otra vez. Se termina la

Tabla 7.32: Pruebas del código adx_pic_tap.c. Tap

TAP

Registro o Variable	Descripción
jacttapstatus	0000 0001 ; bit0=1,detectado Tap en eje z ; ok
jintsource	1100 0011 ; bit6=1,detectado single Tap ; ok ; el resto de 1's son de otras interrupciones que no se usan
accion	DOWN_LEFT
eje_x	23
eje_y	-12 ; (sin zona de reposo implementada) ok
cuadrante	ok
ajustes x,y,z	ok
flag	2
flag_tap	1 ; se entró en la función accion_tap()

prueba con diversos movimientos de inclinación. Tablas 7.33 y 7.34. El funcionamiento es correcto. Se ha comprobado que al realizar los movimientos normales el dispositivo no detecta ningún Tap, por lo que no hay interferencias entre el movimiento y la función tap. También se ha comprobado que al realizar el segundo Tap, la variable numtaps se ha incrementado en una unidad, por lo que el funcionamiento del bucle de la función main es correcto.

Tabla 7.33: Pruebas del código adx_pic_tap.c. Doble Tap 1

TAP num1

Registro o Variable	Descripción
jacttapstatus, jintsource, flag_tap	ok
numtaps	1 ; ok
portA	0x00 ; interrupción borrada pinA0=0 ; ok

Tabla 7.34: Pruebas del código adx_pic_tap.c. Doble Tap 2

TAP num2

Registro o Variable	Descripción
jacttapstatus, jintsource, flag_tap	ok
numtaps	2 ; ok !!!
portA	0x00 ; interrupción borrada pinA0=0 ; ok

IMPORTANTE: Se ha comprobado que si se inicia el dispositivo inclinado en cualquier posición, no sólo en la posición de reposo horizontal habitual, los offsets ajustan los valores de salida de tal manera que el movimiento posterior toma como referencia esa posición inicial particular, y ejecuta las acciones de movimiento teniendo en cuenta esa nueva posición inicial distinta de la horizontal.

Programa: pic_bt.c, módulo Bluetooth y comandos AT

En Linux primero se va a comprobar que la conexión con el RN42 se puede realizar, usando un programa administrador de Bluetooth que funciona bajo la

pila Bluez de Linux. Una vez conectado el dispositivo, el LED del RN42 parpadea quedando en modo descubrimiento (discovery). El administrador Bluetooth del ordenador lo detecta con el nombre de RN42-0977. Al realizar el primer emparejamiento (pairing) utilizamos la contraseña por defecto que es 1234. Al conectarse el LED pasa a luz fija. La conexión se realiza en el puerto rfcomm0.

La dirección MAC del RN42 también la ofrece, y es:

00:06:66:42:09:77

Para visualizar si se está recibiendo algo se abre el programa terminal Moserial, que es un equivalente al Hyperterminal de Windows.

Antes hay que dar permisos, con la siguiente línea de comandos en un terminal:

sudo chmod 666 /dev/rfcomm0

Se han realizado diversas pruebas con los printf() programados en el PIC, hasta llegar a la configuración correcta del printf() según lo que se quería mandar. En una de ellas se mandaban entre otros los valores hexadecimales 0xa1, 0x30, 0x01, 0x00, recibiéndose en Moserial los valores A1 30 01 00 en hexadecimal; ver figura 7.3.

En otra prueba al mandar las palabras UP y DOWN se recibía idem en caracteres ASCII.

También se ha usado el programa Docklight en Windows, que igualmente es un equivalente a Hyperterminal, para hacer pruebas puesto que los programas PIC C Compiler y MPLAB IDE usados para programar el PIC estaban instalados en Windows. Figuras 7.4, 7.5 y 7.6.

Comandos AT

El módulo RN42 puede configurarse y da información propia a través de los comandos AT de los que dispone. Para ello se ha utilizado en Windows el programa Docklight para enviar caracteres al módulo Bluetooth. Éste dispone de 60 segundos desde que es alimentado para entrar en el modo Command y usar los comandos AT.

Enviando los caracteres $$$ se entra al modo command. El módulo responde CMD.

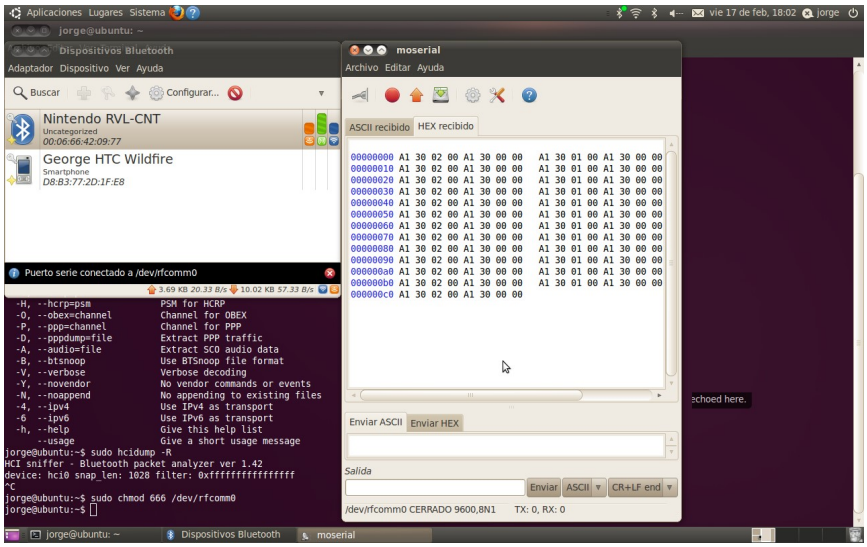


Figura 7.3: Pantalla del programa Moserial. Hexadecimales

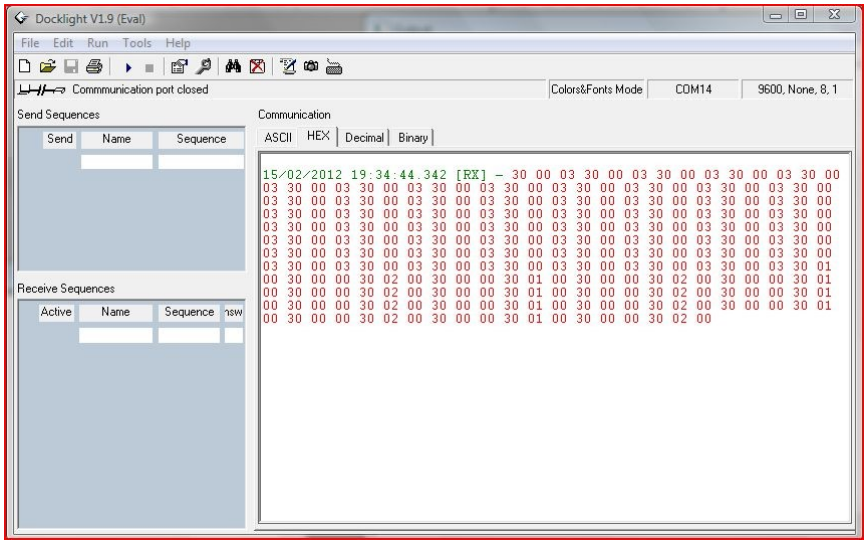


Figura 7.4: Pantalla del programa Docklight 1

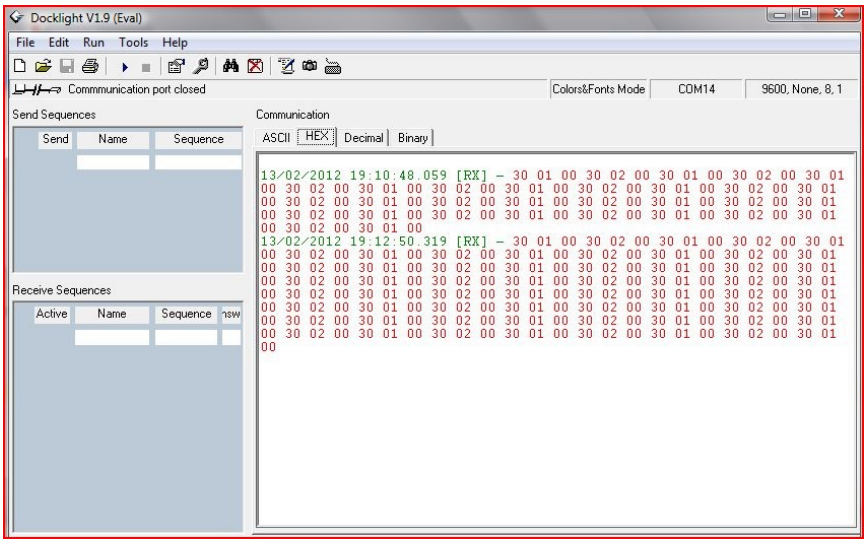


Figura 7.5: Pantalla del programa Docklight 2

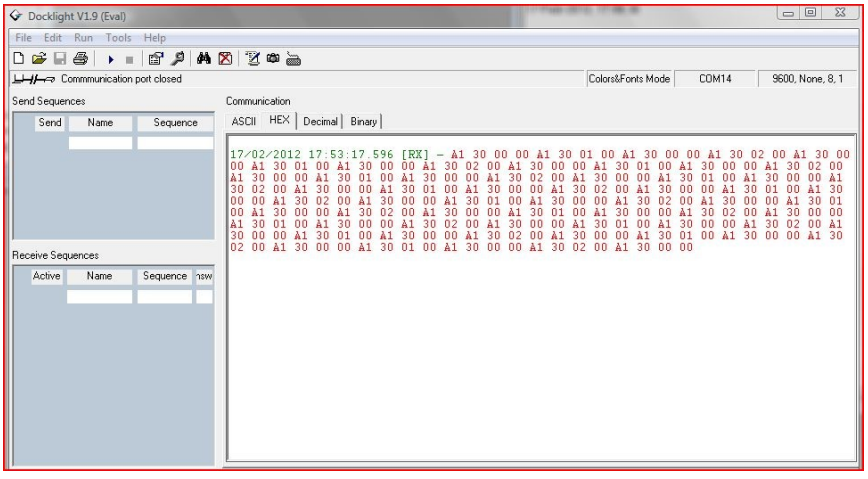


Figura 7.6: Pantalla del programa Docklight 3

Tecleando D y después E, el módulo responde con la información básica y extendida del dispositivo, que se puede ver en la figura 7.7.

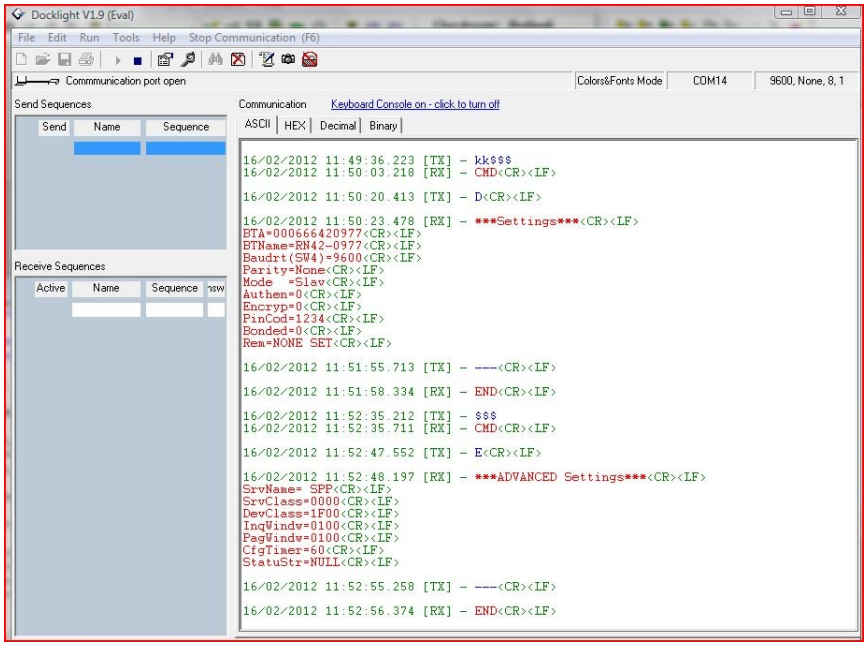


Figura 7.7: Pantalla del programa Docklight. Comandos AT

Los parámetros más importantes son:

- MAC address: 000666420977
- BT-Name: RN42-0977
- Baud Rate: 9600
- PinCod: 1234
- SrvClass: 0000 y DevClass: 1F00 que implica Class: 0x001F00 (hex)

Todos los parámetros excepto el MAC address son configurables. En nuestro caso modificaremos mediante comandos AT el nombre y la clase, ya que nos era imprescindible para conjugar con el programa Cwiid.

Cambio de BT-Name:

El Wiimote^{TM} de Nintendo^{TM} tiene como nombre “Nintendo RVL-CNT-01”, así que se intentó poner ese nombre en el RN42, pero el comando AT asociado tiene un máximo de 20 caracteres incluido el comando, por lo que finalmente se cambió al nombre “Nintendo RVL-CNT”, con este comando:

SN,Nintendo RVL-CNT

El número de clase también se cambió a 0x002504, que es el que trae el Wiimote^{TM}, con estos comandos:

SC,0000 SD,2504

Para salir del modo command se teclea —<enter>.

Programas: pic_bt_pointer.c, adx_pic_bt.c y rn42_pointer.c

Se prueba el código pic_bt_pointer.c programado en el PIC con el programa rn42_pointer.c como programa cliente del PC, y se comprueba que el ciclo cerrado programado en el PIC se reproduce de manera exacta en la pantalla con el cursor del ordenador.

Se prueba el código adx_pic_bt.c programado en el PIC con el programa rn42_pointer.c como programa cliente del PC, y se comprueba de forma satisfactoria que el cursor del ordenador se mueve siguiendo el movimiento de la placa experimental. Aunque en las figuras no se aprecia el movimiento, las figuras 7.8, 7.9 y 7.10, muestran las pantallas en Linux que se ven al ejecutar el programa.

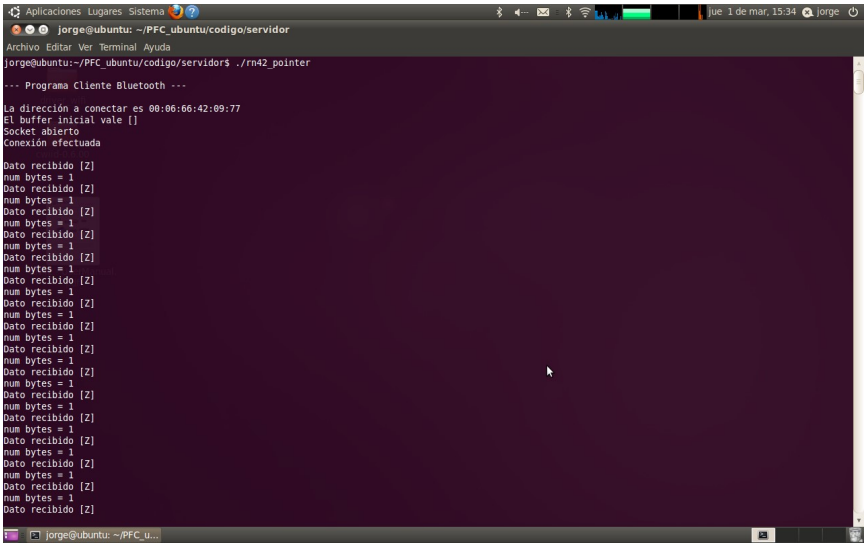


Figura 7.8: Pantalla 1 del terminal en Linux. Programa rn42_pointer.c

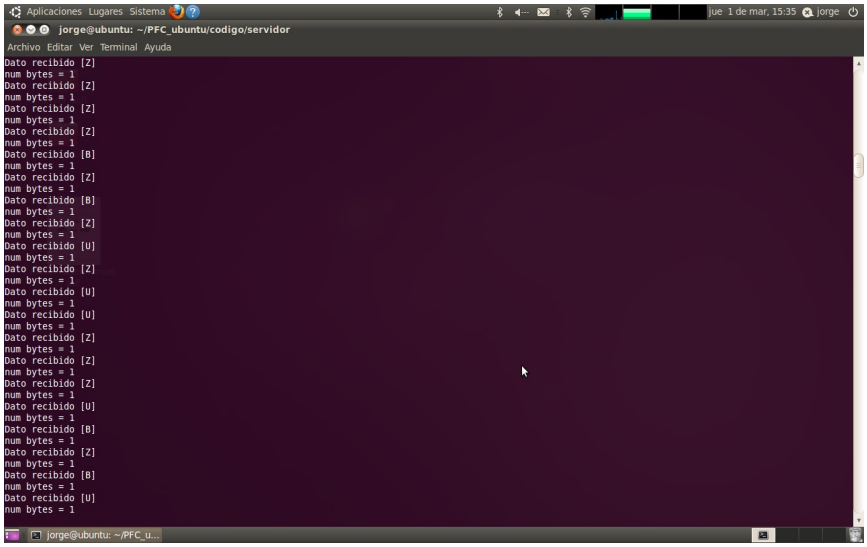


Figura 7.9: Pantalla 2 del terminal en Linux. Programa rn42_pointer.c

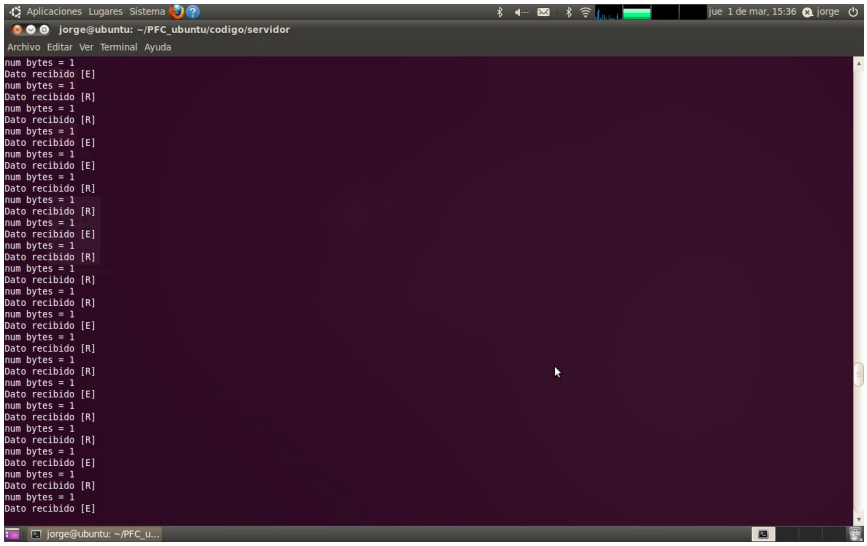


Figura 7.10: Pantalla 3 del terminal en Linux. Programa rn42_pointer.c

Programas: values.c y bt_client.c

Se prueba el código values.c programado en el PIC con el programa bt_client.c como programa cliente del PC.

Al ejecutarse, se ve en pantalla que los valores están expresados en decimal con un rango de aproximadamente (-290, +290) por eje.

El acelerómetro es muy sensible por lo que aún estando en reposo los valores oscilan en un rango de aproximadamente +/- 40 (en decimal). Al mover el dispositivo los valores de cada eje cambian según la dirección y el sentido del movimiento, de acuerdo con lo expresado en la hoja de características del acelerómetro.

En la figura 7.11 se puede ver el aspecto del programa al ser ejecutado. La variación en los valores de cada eje ante cada movimiento concreto se muestra en las figuras 7.12, 7.13, 7.14, 7.15, 7.16 y 7.17, y se comprueba que el funcionamiento es correcto.

Por tanto, el dispositivo cumple su objetivo de trasladar el movimiento de los ejes x, y, z, desde el acelerómetro al ordenador. Funcionamiento correcto.

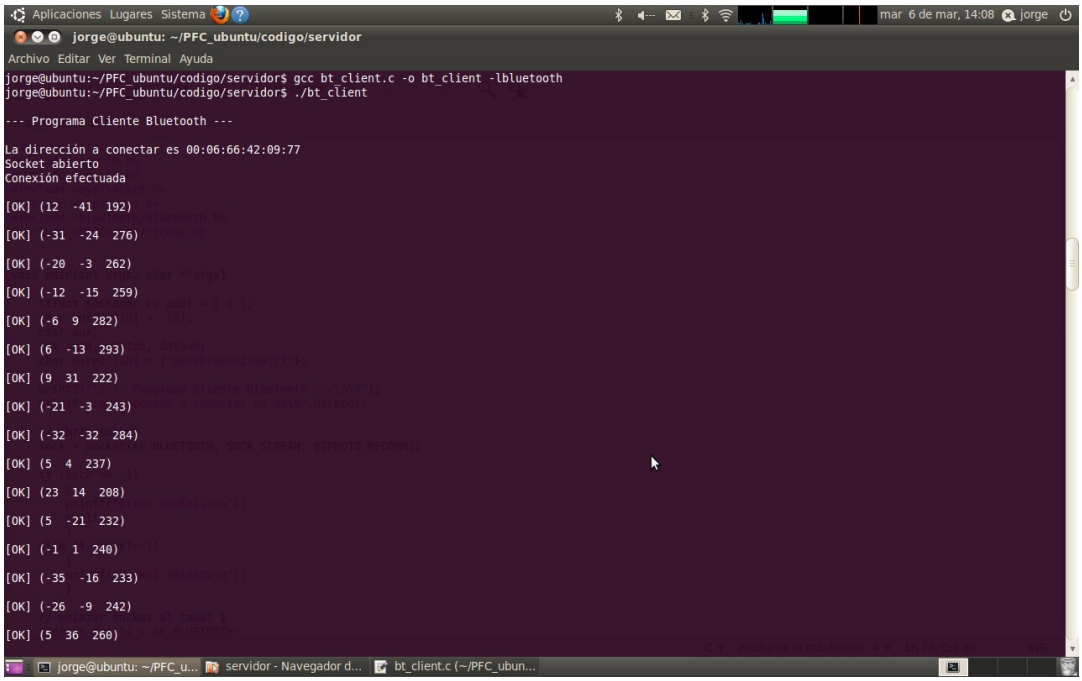


Figura 7.11: Pantalla del terminal en Linux. Programa bt_client.c

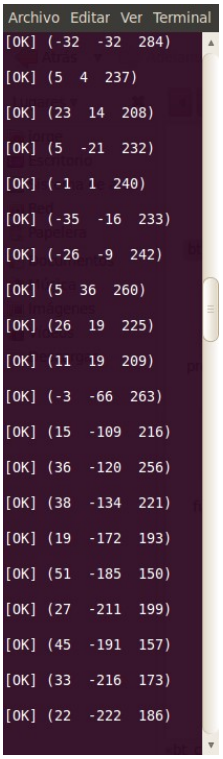


Figura 7.12: Pantalla del terminal en Linux. Programa bt_client.c. Adelante

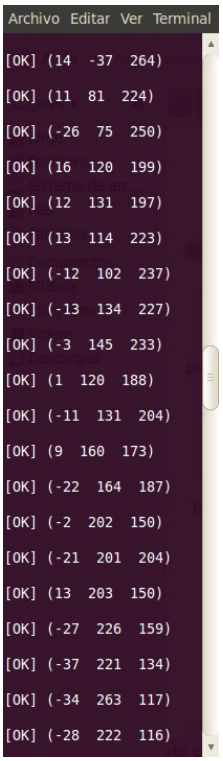


Figura 7.13: Pantalla del terminal en Linux. Programa bt_client.c. Atrás

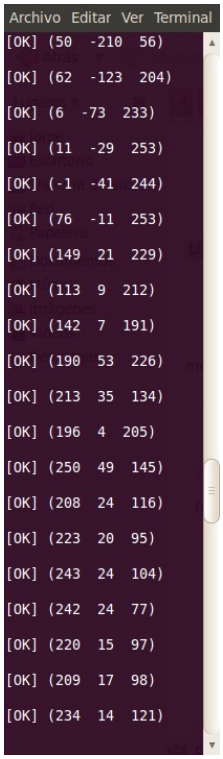


Figura 7.14: Pantalla del terminal en Linux. Programa bt_client.c. Izquierda

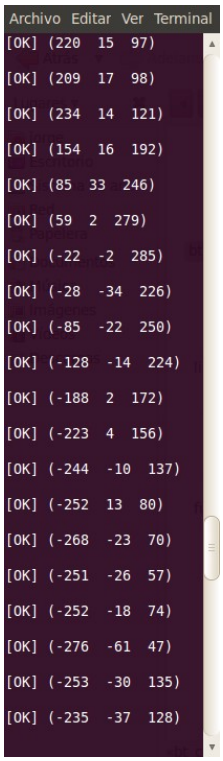


Figura 7.15: Pantalla del terminal en Linux. Programa bt_client.c. Derecha

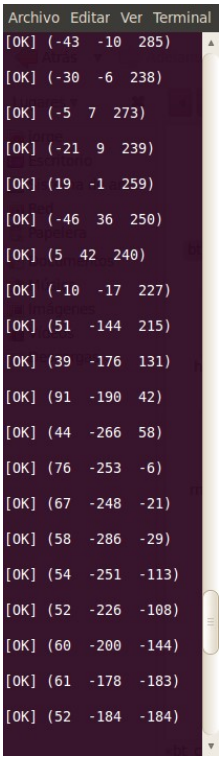


Figura 7.16: Pantalla del terminal en Linux. Programa bt_client.c. Boca abajo

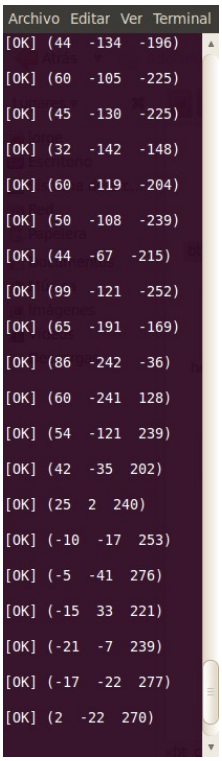


Figura 7.17: Pantalla del terminal en Linux. Programa bt_client.c. De nuevo horizontal

7.2. Ensayos y resultados placa definitiva

Una vez recibida la PCB definitiva con todos los componentes montados se prueba en primer lugar la alimentación. Se introducen las 3 pilas en sus portapilas y se enciende el dispositivo mediante el interruptor deslizante. El LED rojo se enciende en posición fija y el LED verde se enciende con una luz intermitente de la frecuencia esperada ya que el módulo Bluetooth comienza en modo Discovery. Por tanto la alimentación del dispositivo es correcta.

Seguidamente se procede a la programación del PIC. Mediante MPLAB IDE, el programador MPLAB ICD2 y el adaptador RJ11–microUSB, se conecta el ordenador con el dispositivo y se dispone todo para programar el código values.c. La programación se realiza de forma satisfactoria.

Por otro lado, como el componente RN42 de esta PCB definitiva es distinto al que se utilizó en la placa experimental, hay que tener en cuenta que el nombre y la dirección MAC será distinta. Efectivamente, al encender la antena Bluetooth del ordenador y al encender el dispositivo, Linux lo detecta, y muestra la siguiente identificación del módulo Bluetooth detectado:

Nombre: RN42–34E2

MAC address: 00:06:66:4B:34:E2

Por tanto, como la dirección MAC es la que marca a quién se conecta el programa bt_client.c, hay que modificar este código para sustituir esta dirección nueva por la anterior. El nuevo código se llama bt_client_def.c, que es exactamente igual a bt_client.c, salvo por la dirección MAC. Una vez modificado, hay que compilar de nuevo en el terminal de Linux para generar el objeto ejecutable, con la siguiente línea:

```
gcc bt_client_def.c -o bt_client_def -lbluetooth
```

Se ejecuta el programa desde la ruta donde se encuentra el objeto ejecutable con la línea:

```
./bt_client_def
```

Se enciende el dispositivo y el programa se conecta al módulo Bluetooth. Ver figura 7.18. En este momento el LED verde pasa a tener luz fija indicando

conexión. Entonces, el programa comienza a mostrar los valores de los 3 ejes del acelerómetro con el formato:

[OK] (eje_x eje_y eje_z)

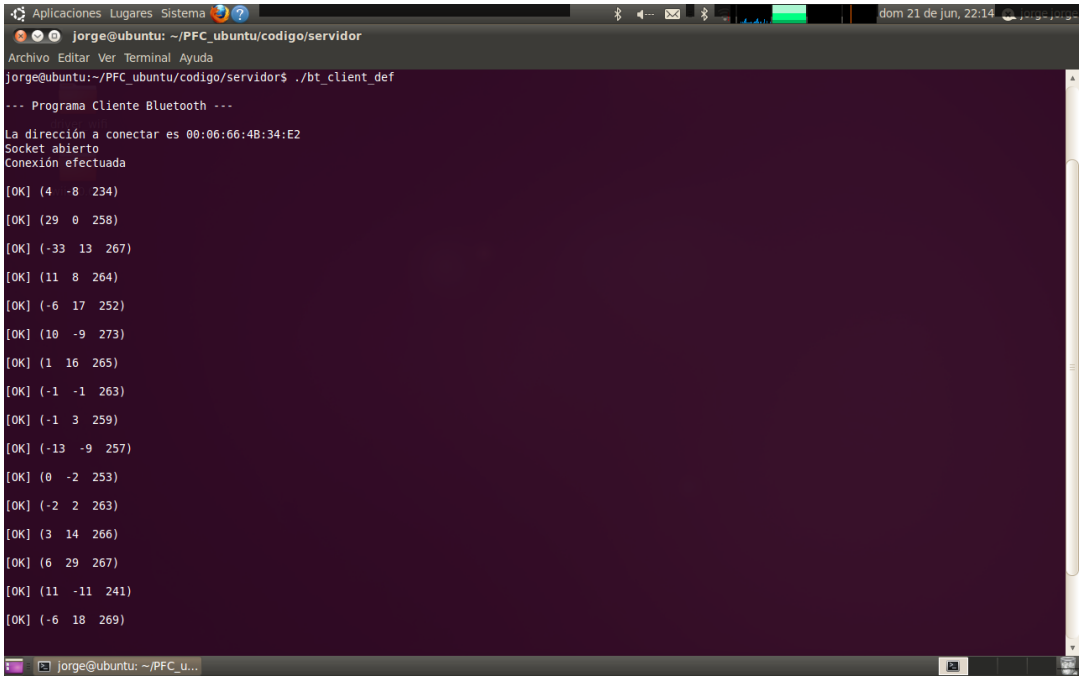


Figura 7.18: Pantalla del terminal en Linux. Programa bt_client_def.c. Conexión con el dispositivo y primeros valores

Moviendo el dispositivo en todas las inclinaciones posibles se comprueba cómo los valores de los ejes van cambiando de acorde a lo esperado. Por tanto, el funcionamiento es correcto. El dispositivo traslada de forma inalámbrica los datos de movimiento al ordenador. Ver figura 7.19.

7.3. Integración

El último proceso consiste en integrar todo el hardware de la interfaz. Con el soporte y la tapa ya obtenidos a través de la impresora 3D, hay que introducir la PCB con los componentes montados en el hueco del soporte habilitado a tal efecto. Se introducen las 3 pilas botón en sus respectivos portapilas y se cierra el soporte con la tapa. Para fijar el conjunto se utilizan 4 tornillos de carpintería avellanados de dimensiones 2 x 12 mm de la marca comercial Spax; ver figura 7.20.

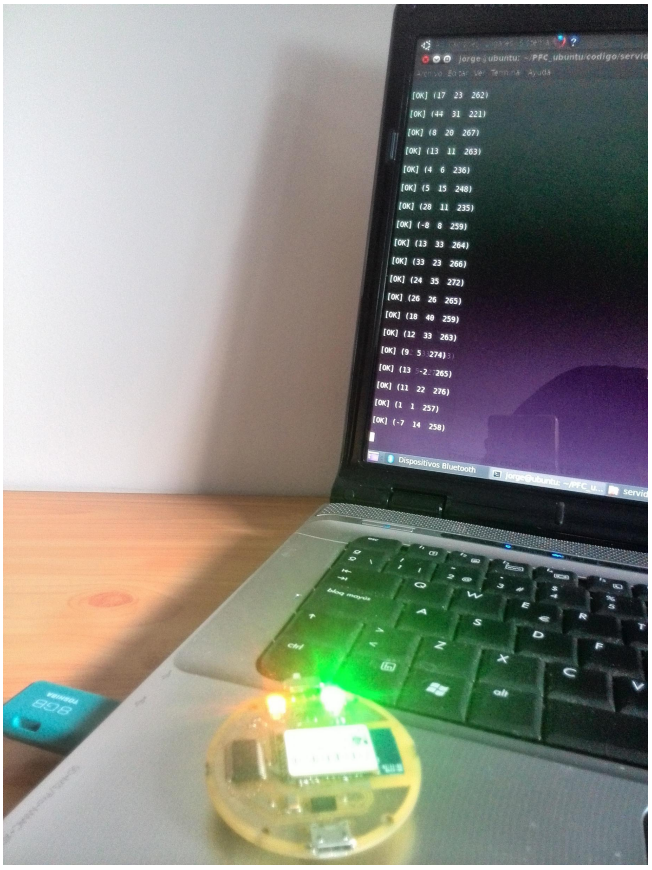


Figura 7.19: Pantalla del terminal en Linux. Programa bt_client_def.c. Dispositivo conectado al ordenador enviando valores de los 3 ejes

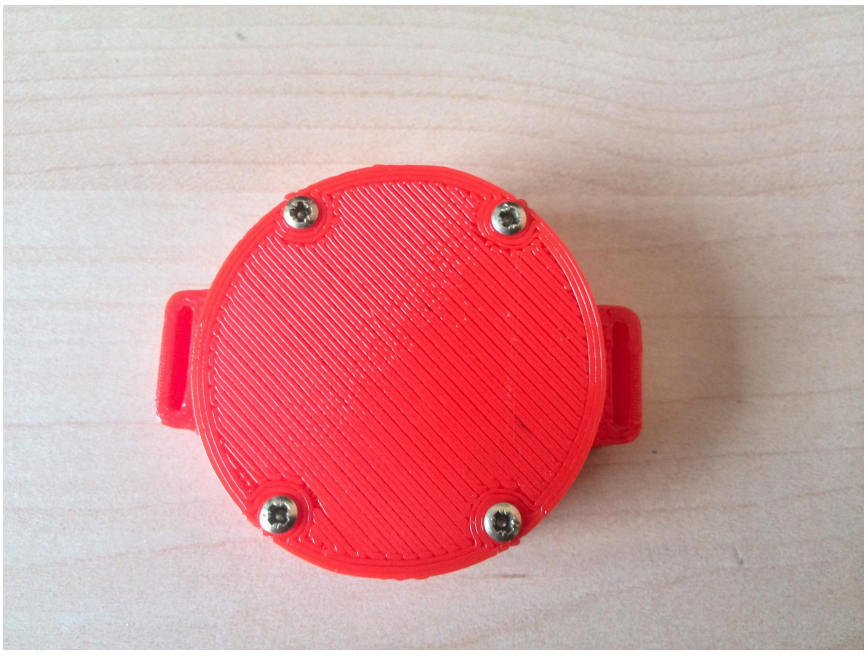


Figura 7.20: Integración de PCB, soporte y tapa con tornillos de fijación

Para acoplar la interfaz en el brazo de la persona se utiliza una correa de velcro a medida que se puede ver en la figura 7.21. Así, el movimiento de la interfaz es solidario al movimiento de la extremidad de la persona.

Figura 7.21: Interfaz acoplada al brazo mediante correa de velcro

Capítulo 8

Conclusiones y desarrollos futuros

Tras todo el proceso de desarrollo hardware, software y mecánico, se ha llegado a conseguir fabricar y a hacer funcionar la interfaz.

Los códigos definitivos completos para el microcontrolador y para el PC, así como los scripts o códigos escritos en OpenSCAD que determinan los diseños definitivos del soporte se pueden ver en `https://github.com/legageorge/Interfaz-micromote`.

En primer lugar, la PCB con los componentes electrónicos integrados tiene un tamaño lo suficientemente pequeño como para dar por válido el objetivo de conseguir un dispositivo pequeño y manejable. Se han reducido al máximo las dimensiones respecto de interfaces anteriores, resultando un dispositivo que se puede acoplar fácilmente a cualquier persona. Asimismo el peso del conjunto es muy ligero y por tanto esa no es una característica que pueda condicionar su utilización.

Se ha conseguido un consumo bajo que implica una autonomía suficiente para funcionar. Por otro lado, el coste por unidad de cada dispositivo en sí mismo (sin considerar los costes de personal) es bastante asequible, por lo que el precio no es un obstáculo para escoger esta interfaz.

La facilidad de uso que se pretendía parece conseguida en cuanto a que basta con mover el dispositivo de la manera deseada para que esta información de movimiento se traslade al ordenador inmediatamente, lo que implica que efectivamente se cumple el propósito principal de la interfaz, que es la comunicación inalámbrica de los datos de movimiento espacial de la persona al ordenador, cumpliendo la labor de interfaz hombre–máquina.

También señalar que se consigue universalidad en el uso de la interfaz al utilizar tecnología Bluetooth ya que cualquier ordenador puede recibir ese tipo de datos, ya sea con una antena Bluetooth integrada o con un periférico que se puede añadir fácilmente.

Por tanto, los objetivos principales del proyecto que se detallaban en el Capítulo 1 se han conseguido. Y se puede decir, que respecto de otras posibles soluciones tecnológicas que puedan competir con esta (por ejemplo, soluciones con Arduino), la interfaz μMote es un sistema dedicado, no generalista, y que por tanto tiene sólo la electrónica estrictamente necesaria para el objetivo concreto, consiguiéndose así un tamaño y peso menores.

8.1. Análisis crítico

Hay que tener en cuenta que la funcionalidad de esta interfaz se reduce a trasladar el movimiento de los tres ejes espaciales según la inclinación del dispositivo al ordenador. La transferencia de los datos al robot se explica en el siguiente apartado puesto que es una ampliación a desarrollar después de este proyecto. Para controlar un robot, y en concreto el robot ASIBOT, son necesarias más actuaciones aparte del movimiento en 3 ejes que proporciona el trabajo realizado. ASIBOT tiene 5 grados de libertad por lo que faltan dos parámetros para manejar todo el movimiento del robot. Pero aparte del movimiento en sí mismo, el robot requiere de más órdenes y/o parámetros para realizar sus tareas.

En el diseño software de la interfaz se ha mostrado la función Tap y Doble Tap del acelerómetro, que a modo de click de ratón por aceleración brusca del dispositivo, permitía disponer de 2 parámetros más. Sin embargo, las pruebas experimentales para esta característica no fueron totalmente satisfactorias, por lo que se decidió dejarlo sin utilizar y centrarse sólo en el movimiento. Ya se comentó anteriormente y se va a describir en el siguiente apartado, un trabajo futuro basado en pulsadores complementarios que permita completar la funcionalidad para manejar el robot.

Por otra parte, el diseño mecánico se ha centrado sólo en poder acoplar el dispositivo a la persona a modo de reloj de pulsera, y de esta manera limitando el perfil del usuario. Para personas discapacitadas que no puedan mover los brazos o piernas este soporte no vale, por lo que habría que diseñar otro. Eso sí, la PCB sería la misma ya que es capaz de adaptarse a distintas posiciones iniciales y por ello ir acoplada a cualquier parte del cuerpo.

8.2. Ampliaciones

La primera ampliación necesaria a realizar es el enlace del programa cliente del ordenador que recibe los datos del movimiento de los 3 ejes, con la arquitectura software del robot, en este caso el robot ASIBOT [1]. Hay que escribir un código para incluir los tres parámetros de movimiento en la arquitectura YARP de ASIBOT, y asignar cada uno de ellos al movimiento de uno de los ejes del robot; ver figura 8.1. De esta manera, la interfaz sí consigue servir de nexo de comunicación entre la persona y el robot.

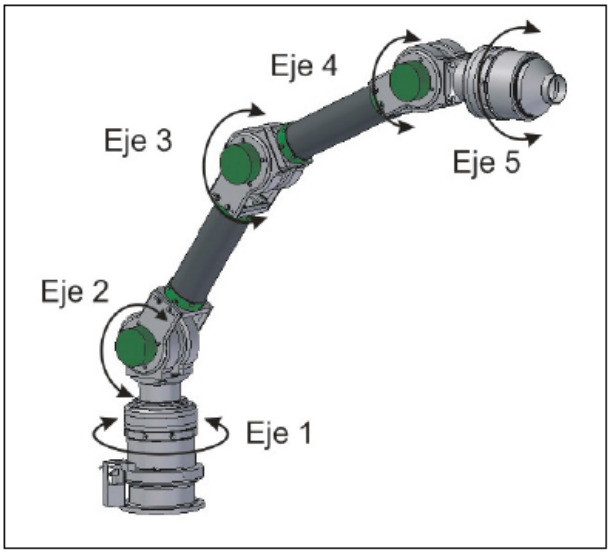


Figura 8.1: Configuración cinemática del robot ASIBOT

Para completar el control de todos los ejes se puede desarrollar un sistema de pulsadores externos a la interfaz, que comuniquen directamente con el robot, y que se encarguen de los otros dos parámetros de movimiento que restan. Serían 2 pulsadores dobles para recorrer el rango de valores de movimiento de cada eje en sentido ascendente o descendente. Una idea inicial es usar el movimiento de los tres ejes de la interfaz para controlar los ejes 1, 2 y 3 del robot, ya que de una manera rápida e intuitiva el robot se colocaría en la posición general que se pretende. Y dejar el control de los ejes 4 y 5 a los pulsadores, para tener con ellos un ajuste fino de la posición final del extremo del robot. Otra alternativa, para no recurrir a los pulsadores, es utilizar la interfaz inicialmente para controlar el grupo de ejes 1, 2 y 3, y utilizar la característica de TAP de la interfaz para cambiar al control del grupo de ejes 4 y 5. Es decir, el TAP o aceleración brusca de la interfaz, serviría para intercambiar el control del movimiento entre los grupos de ejes 1–2–3 y 4–5. De esta manera, la interfaz sería totalmente independiente para controlar el movimiento del robot en su totalidad.

8.3. Desarrollos futuros

La interfaz se puede mejorar desarrollando futuras versiones, que tomando como base el dispositivo actual, puedan revisar los siguientes aspectos.

El tamaño. El módulo Bluetooth del dispositivo es bastante grande en comparación con el resto de componentes, llegando a ocupar prácticamente la cuarta parte de la PCB. Encontrando un componente Bluetooth significativamente más pequeño se puede reducir bastante el diámetro final en un diseño circular. Por otro lado, el conjunto de las tres baterías botón con sus portapilas, resultan bastante voluminosos, y aunque no penalizan el diámetro al estar en la cara BOTTOM de la PCB, sí se podría reducir mucho la altura total si se encontrara otra alternativa, por ejemplo pilas botón más delgadas que mantengan la autonomía. Esto considerando mantener el sistema de alimentación por pilas desechables. Se puede optar por una batería delgada recargable aunque esto traería el problema de añadir un circuito de carga (que precisamente fue lo que inclinó la decisión de optar por una alimentación con pilas).

El consumo y la autonomía. La mayor parte del consumo del dispositivo es debido al módulo Bluetooth. Sustituyendo el componente actual por otro de mucho menos consumo ya se reduciría de forma sustancial el consumo total y se tendría mucha mayor autonomía. O bien, manteniendo la autonomía, al consumir menos se podrían escoger unas pilas más delgadas y/o más pequeñas para reducir el tamaño.

La programación. Se puede añadir al software la capacidad de reconocer diversos patrones de movimiento, con la intención de poder realizar tareas concretas preestablecidas del robot. Dejando, por ejemplo, al inicio de la conexión del dispositivo una ventana de tiempo suficiente, si se realiza un patrón de movimiento determinado que vaya asociado a una tarea, el robot ejecuta la secuencia completa de movimientos que requiera esa tarea. Así, se puede crear una biblioteca de movimientos espaciales que tengan asignada una tarea concreta para el robot. Por supuesto, el dispositivo en modo normal seguiría ejerciendo de controlador de movimiento libre. Integrando esta idea con los pulsadores externos, se puede llevar a la interfaz a dos modos de funcionamiento: Modo Libre y Modo Tarea. Y así, poder elegir si se quiere mover el robot de forma libre controlándolo con tu propio movimiento, o bien si se quiere hacer uno de los movimientos preestablecidos de la biblioteca para que el robot realice una tarea determinada.

El soporte mecánico. Sería importante hacer un estudio de los distintos perfiles de usuarios de la interfaz. En las personas discapacitadas la diversidad de patologías es muy grande. Aún teniendo en cuenta que esta interfaz se ideó para personas con falta de movilidad en las manos que no pueden manejar un ratón,

joystick, PDA o similar, el diseño se ha limitado para ser colocado en una extremidad. Pues bien, la interfaz debe de ser flexible para todo tipo de casuística, pues hay discapacidades en las que no pueden mover las extremidades. Por tanto, habría que diseñar soportes y sistemas de sujeción para todos los casos, a saber, pulseras para brazos y/o piernas, cinta para cabeza, cinta para tórax, etc. Aquí es clave la consulta y opinión de los usuarios y en todo caso se intenta que el uso de la interfaz sea sencillo y amigable, y por ello, buscar soluciones tipo gorra, gafas, auricular y otros, todos ellos con la PCB integrada. Señalar de nuevo que se comprobó en los ensayos que el dispositivo puede comenzar en cualquier posición inicial, no necesariamente la horizontal, y a partir de ahí mandar los valores relativos de inclinación respecto de esa posición inicial.

Otro futuro desarrollo de integración de la interfaz con el robot sería el control de un menú gráfico instalado en una tablet del usuario, de tal manera que el movimiento de la persona detectado por el dispositivo acoplado haga moverse por el menú gráfico, y con la ayuda de pulsadores externos o voz ir confirmando cada selección. Así, se puede conseguir una navegación por el entorno gráfico del robot, en el que la interfaz haría la función de mando o ratón inalámbrico.

Por último, se puede llevar más allá el concepto de interfaz, cambiando el objetivo orientado a personas llamado interfaz hombre–máquina (HMI) y llevándolo al concepto de interfaz máquina–máquina (M2MI).[1] La idea es utilizar esta interfaz como detector de movimientos de robots móviles para que puedan comunicarse entre sí, y puedan conocer en tiempo real los movimientos que están realizando sus compañeros. Es decir, se puede acoplar la interfaz a robots ya existentes, y el movimiento que efectúen esos robots sería enviado vía Bluetooth, o bien a un ordenador que centralice todos los movimientos, o bien, a dispositivos receptores de otros robots que habría que desarrollar. Esto permite conocer movimientos de inclinación o giros respecto de ejes, no trayectorias cartesianas. Se puede dar la situación en que grupos de robots autónomos necesitaran conocer entre ellos sus movimientos de giro para realizar una tarea común. Interfaces adaptadas y acopladas en ellos podrían dar solución a esa necesidad.

[1] M2M, del inglés, machine–to–machine. No confundir con MMI, man–machine interface, equivalente a HMI, human–machine interface.

Bibliografía

[1] A. Jardón Huete, *Metodología de diseño de robots asistenciales. Aplicación al robot portátil ASIBOT*. PhD thesis, Universidad Carlos III de Madrid, 2006.

[2] V. López Vaquero, "Dispositivo inalámbrico para facilitar el acceso al ordenador." Proyecto Fin de Carrera. Universidad Carlos III de Madrid, 2008.

[3] T. Oetiker, "The not so short introduction to latex 2e." GNU General Public License, 2011.

[4] N. Reid, *Wireless Mobility: The why of wireless*. Mc Graw-Hill, 2010.

[5] W. Wheeler, *Integrating Wireless Technology in the enterprise*. Digital Press, 2004.

[6] V. Hernández Muñoz, "Desarrollo de una aplicación móvil en sistema android para el control remoto de dispositivos mediante tecnología bluetooth 4.0." Proyecto Fin de Carrera. Universitat Politecnica de Valencia, 2014.

[7] D. Green and S. Goldner, "Remote control system for a video computer game," July 30 1985. US Patent 4,531,740.

[8] M. Escobosa, Y. Yun, and F. Chen, "Wireless control and pointer system," Aug. 7 2001. US Patent 6,271,831.

[9] J. Baichtal, *Robot Builder: The Beginner's guide to building robots*. Que, 2014.

[10] C. M. Durán and R. A. Castro, "Comunicación inalámbrica basada en tecnología bluetooth para la automatización de procesos industriales." Revista El Hombre y la Máquina num 39. Universidad Autónoma de Occidente. Colombia, 2012.

[11] R. Howard, "Wireless control device," June 8 2004. US Patent 6,747,632.

[12] S. Ferrer Marqués, "Recursos digitales para educación especial." `http://www.ardilladigital.com/tecnologia_educativa.htm`. Último acceso: junio 2015.

[13] T. Moya, J. Goglino, and D. Hoyos, "Red de sensores y control inalámbrico para un sistema de generación de vapor solar térmico." Avances en energías renovales y medio ambiente, 2010.

[14] J. González Víctores, A. Jardón, S. Morante, F. Bonsignorio, and C. Balaguer, "Give me the red can: Asibot assistive robot platform task creation through multi-modal interaction." Conference: International Congress on Design, Research Networks, and Technology for all - DRT4ALL, 2013.

[15] O. Romero Granados, "Ratón bluetooth para personas con discapacidad." Proyecto Fin de Carrera. Universitat Politécnica de Catalunya, 2006.

[16] A. Jardón Huete, V. López Vaquero, and C. Balaguer, "Dispositivo inalámbrico para facilitar el acceso al ordenador." Congreso Internacional sobre Domótica, Robótica y Teleasistencia para Todos DRT4LL, 2009.

[17] L. Romero Bachiller, "Diseño, desarrollo y manual de una interfaz gráfica distribuida para maggie." Proyecto Fin de Carrera. Universidad Carlos III de Madrid, 2010.

[18] A. Sánchez García and M. López Montesinos, "Wireless devices in nursing education." Invest Educ Enferm, 2013.

[19] R. Palencia López, "Migración de la plataforma a bordo del robot asistencial asibot." Proyecto Fin de Carrera. Universidad Carlos III de Madrid, 2009.

[20] R. Rodrigo Valero, "Hmi wiimote: La integración del mando inalámbrico wii remote dentro de la arquitectura software yarp." Proyecto Fin de Carrera. Universidad Carlos III de Madrid, 2010.

[21] D. García Sánchez, "Sns xsens: La integración del sensor inercial mti dentro de la arquitectura software yarp." Proyecto Fin de Carrera. Universidad Carlos III de Madrid, 2010.

[22] A. Torres, S. Gamboa, *et al.*, "Diseño de un mouse óptico facial para pacientes que presentan discapacidad parapléjica." Revista cubana de física vol.27, 2010.

[23] M. Bureau, J. Azkoitia, G. Ezmendi, *et al.*, "Non-invasive, wireless and universal interface for the control of peripheral devices by means of head movements," *10th international conference on rehabilitation robotics of the IEEE*, 2007.

[24] J. González Víctores, *Robot Imagination System.* PhD thesis, Universidad Carlos III de Madrid, 2014.

[25] J. González Víctores, "Software engineering techniques applied to assistive robotics: guidelines and tools," Master's thesis, Universidad Carlos III de Madrid, 2010.

[26] C. Escolano and J. Mínguez, "Sistema de teleoperación multi-robot basado en interfaz cerebro-computador," *Revista iberoamericana de automática e informática industrial*, 2011.

[27] Y.-T. Liao, W. Biederman, and B. Otis, *Low Power emerging wireless technologies*. CRC Press, 2013.

[28] M. Arenas Mas, "Diseño e implementación de un sistema de adquisición de aceleraciones con procesamiento mediante controlador." Proyecto Fin de Carrera. Universidad de Sevilla, 2008.

[29] T. B. Jones and N. G. Nenadic, *Electromechanic and MEMS*. Cambridge University Press, 2013.

[30] D. Monje Centeno, "Conceptos electrónicos en la medida de la aceleración y la vibración." Trabajo Final de Curso. Universidad de Sevilla, 2010.

[31] L. Jin-Shyan, S. Yu-Wei, and S. Chung-Chou, "A comparative study of wireless protocols: Bluetooth, uwb, zigbee and wi-fi," *33rd annual conference of the IEEE*, 2007.

[32] C. F. S. Jennifer Bray, *Bluetooth 1.1: Connect without cables*. Prentice Hall, 2001.

[33] D. Barbolla Asenjo, "Desarrollo de una aplicación de comunicación bluetooth entre una pda y sensores." Proyecto Fin de Carrera. Universidad Carlos III de Madrid, 2009.

[34] V. Garg, *Wireless Communications and Networking*. Morgan Kaufmann, 2010.

[35] O. F. Corredor, L. F. Pedraza, and C. A. Hernández, "Tecnología bluetooth: Alternativa para redes celulares de voz y datos." Revista Visión Electrónica. Colombia, 2009.

[36] E. de Gracia Corrales, "Pasarela modular de comunicaciones entre dispositivos inalámbricos y el sistema domótico x10," Master's thesis, Universidad Rey Juan Carlos de Madrid, 2009.

[37] `http://www.prometec.net/bt-hc05/`. Último acceso: junio 2015.

[38] "Tienda online rs components." `http://es.rs-online.com`. Último acceso: junio 2015.

[39] *Microchip MPLAB ICD2 In-Circuit Debugger User's Guide*.

[40] M. de prácticas. Universidad Carlos III de Madrid, *Diseño electrónico asistido por ordenador*.

[41] *Microchip MPLAB IDE User's Guide with MPLAB editor and MPLAB SIM simulator*.

[42] A. C. López, *Manual de usuario del compilador PCW de CCS*.

[43] N. Gardner, *PIC micro MCU C. An introduction to programming The Microchip PIC in CCS C*.

[44] *Microchip In-Circuit Serial Programming (ICSP) Guide*.

[45] "Programación básica de sockets en unix." `http://es.tldp.org/Tutoriales/PROG-SOCKETS/prog-sockets.html`. Último acceso: junio 2015.

[46] "Bluetooth programming in c with bluez." `http://people.csail.mit.edu/albert/bluez-intro/x559.html`. Último acceso: junio 2015.

Apéndice A

Presupuesto

A.1. Datos identificativos

1) Nombre del proyectista

   Jorge Jiménez Rodríguez.

2) Departamento

   Departamento de Ingeniería de Sistemas y Automática.
   Universidad Carlos III de Madrid.

3) Descripción del proyecto

   Título: μMote: Interfaz hombre–máquina inalámbrica de control para robótica asistencial.

   Duración: El desarrollo del proyecto ha necesitado una dedicación de 11 meses, contados como dedicación exclusiva y de manera correlativa.

4) Presupuesto total del proyecto

   El presupuesto del proyecto asciende a la cantidad total de 35827,45 euros (€), IVA incluido.

A.2. Desglose presupuestario

Se desglosa el presupuesto de la siguiente manera:

1) Costes de personal

   Tabla de presupuesto de personal; tabla A.1.

2) Costes de material

   Tabla de presupuesto de la placa experimental (tabla A.2) y tabla de presupuesto de la placa definitiva (tabla A.3).

3) Costes de equipos

   Tabla de presupuesto de equipos utilizados; tabla A.4.

4) Subcontratación de tareas

   Tabla de presupuesto de subcontratación; tabla A.5.

A continuación se presentan las tablas anteriormente citadas.

Tabla A.1: Tabla de presupuesto de personal

Apellidos, Nombre	Categoría	Dedicación [meses]	Coste hombre/mes [€] (1)	Subtotal [€]
Jiménez Rodríguez, Jorge	Ingeniero Técnico	11	3200	35200
TOTAL [€]				**35200**

(1) Estimación de horas: 8 horas diarias x 20 días/mes = 160 horas/mes

(1) Estimación de coste unitario por hora: 20 euros/hora

Tabla A.2: Tabla de presupuesto de la placa experimental

Cantidad	Componente	Valor / Fabricante	Precio unitario [€]	Subtotal [€]
1	Placa de Evaluación incl. ADXL345	Analog Devices	32,59	32,59
1	Microcontrolador PIC16LF876A	Microchip	5,4	5,4
1	Módulo Bluetooth RN42	Roving Networks	12,1	12,1
1	Regulador de tensión TPS62203	Texas Instrument	2,26	2,26
1	Oscilador Cristal	4 Mhz	0,5	0,5
1	Diodo LED	rojo	0,1	0,1
1	Condensador	10 uF	2,6	2,6
1	Condensador	4,7 uF	1,1	1,1
4	Condensador	100 nF	0,18	0,72
2	Condensador	15 pF	0,5	1
1	Resistencia	10 k	0,1	0,1
1	Resistencia	1 k	0,1	0,1
1	Resistencia	220	0,1	0,1
1	Resistencia	10	0,1	0,1
1	Inductancia	10 uH	0,14	0,14
1	Conector RJ11 hembra	Wurth Electronic	0,62	0,62
1	Clema 2 vías	-	0,15	0,15
1	Oscilador Cristal (fase2) SG8002DC	4 Mhz / Epson	6,04	6,04
1	Condensador (fase2)	100 nF	0,18	0,18
1	Placa microUSB hembra (fase2)	Sparkfun	2,25	2,25
1	Conector microUSB macho (fase2)	FCI	0,7	0,7
1	Conector RJ11 hembra (fase2)	Wurth Electronic	0,62	0,62
TOTAL [€] (IVA incl.)				**69,47**

Tabla A.3: Tabla de presupuesto de la placa definitiva

Cantidad	Componente	Valor / Fabricante	Precio unitario [€]	Subtotal [€]
1	Acelerómetro digital ADXL345	Analog Devices	5,62	5,62
1	Microcontrolador PIC16LF876A	Microchip	5,4	5,4
1	Módulo Bluetooth RN42	Roving Networks	12,1	12,1
1	Regulador de tensión TPS62203	Texas Instrument	2,26	2,26
1	Oscilador Cristal SG310SCF	4 Mhz / Epson	3,05	3,05
1	Diodo LED rojo HSMC-A431-X90M1	Avago Technologies	0,31	0,31
1	Diodo LED verde HSMM-A430-X90M2	Avago Technologies	0,96	0,96
1	Condensador (huella 0603)	10 uF	0,18	0,18
1	Condensador (huella 0402)	4,7 uF	0,06	0,06
5	Condensador (huella 0402)	100 nF	0,01	0,05
1	Resistencia (huella 0402)	10 k	0,03	0,03
1	Resistencia (huella 0402)	1 k	0,03	0,03
2	Resistencia (huella 0402)	220	0,03	0,06
1	Resistencia (huella 0402)	10	0,14	0,14
1	Inductancia (huella 0603)	10 uH	0,11	0,11
1	Conector microUSB hembra	TE Connectivity	0,99	0,99
1	Interruptor deslizante MMP121-R	Knitter Switch	0,85	0,85
3	Portapilas botón para PCB	2996 / Keystone	0,86	2,58
3	Batería tipo botón SR44	RS	1,66	4,98
TOTAL [€] (IVA incl.)				**39,76**

Tabla A.4: Tabla de presupuesto de equipos utilizados

Descripción	Coste [€] (C)	% uso dedicado al proyecto [tanto por uno] (D)	Dedicación [meses] (A)	Período de depreciación [meses] (B)	Coste imputable [€] (1)
Fuente de alimentación del laboratorio	590	0,05	4	60	1,97
Programador PIC Microchip	295	0,5	6	60	14,75
Impresora 3D	399	0,05	2	60	0,67
Ordenador portátil con bluetooth	700	1	11	60	128,33
TOTAL [€]					**145,72**

(1) Coste imputable = (A / B) * C * D

Tabla A.5: Tabla de presupuesto de subcontratación

Descripción	Empresa	Coste [€]
Montaje PCB (unidad)	Madritronic S.L.	122,50
Máscara de soldadura	Madritronic S.L.	250,00
TOTAL [€]		**372,50**

A.3. Resumen de costes

Tabla de resumen de costes; tabla A.6.

Tabla A.6: Tabla de resumen de costes

Presupuestos de Costes	Coste [€]
Personal	35200,00
Material 1	69,47
Material 2	39,76
Equipos	145,72
Subcontratación	372,50
TOTAL [€]	**35827,45**

Apéndice B

Listado de componentes

B.1. Placa experimental

LISTADO DE COMPONENTES PLACA EXPERIMENTAL

Cantidad	Componente	Valor / Fabricante	Dimensiones [mm] (ancho x largo x alto)	Montaje	Precio [€]
1	Placa de Evaluación incl. ADXL345	Analog Devices	20 x 20 x 1	pasante	32,59
1	Microcontrolador PIC16LF876A	Microchip	7,49 x 35,18 x 3,43	pasante	5,4
1	Módulo Bluetooth RN42	Roving Networks	13,4 x 25,8 x 2	superficial	12,1
1	Regulador de tensión TPS62203	Texas Instrument	1,6 x 2,9 x 1,15	superficial	2,26
1	Oscilador Cristal	4 Mhz	-	pasante	0,5
1	Diodo LED	rojo	-	pasante	0,1
1	Condensador	10 uF	-	pasante	2,6
1	Condensador	4,7 uF	-	pasante	1,1
4	Condensador	100 nF	-	pasante	0,18 / ud
2	Condensador	15 pF	-	pasante	0,5 / ud
1	Resistencia	10 k	-	pasante	0,1
1	Resistencia	1 k	-	pasante	0,1
1	Resistencia	220	-	pasante	0,1
1	Resistencia	10	-	pasante	0,1
1	Inductancia	10 uH	-	superficial	0,14
1	Conector RJ 11 hembra	Wurth Electronic	-	pasante	0,62
1	Clema 2 vías	-	-	pasante	0,15
1	Oscilador Cristal (fase2) SG8002DC	4 Mhz / Epson	6,6 x 13,7 x 5,3	pasante	6,04
1	Condensador (fase2)	100 nF	-	pasante	0,18
1	Placa microUSB hembra (fase2)	Sparkfun	-	pasante	2,25
1	Conector microUSB macho (fase2)	FCI	-	pasante	0,7
1	Conector RJ 11 hembra (fase2)	Wurth Electronic	-	pasante	0,62

B.2. Placa definitiva

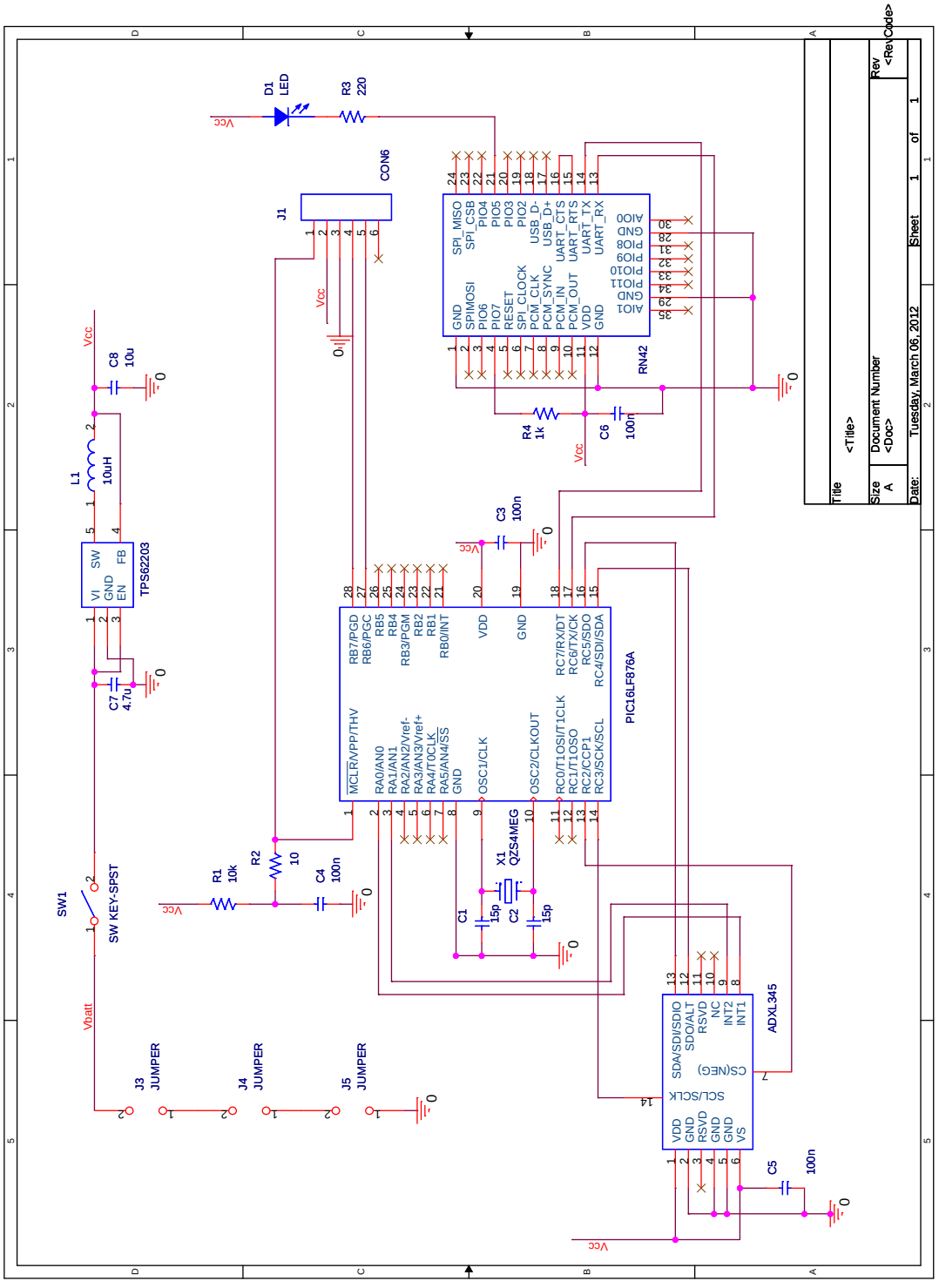
LISTADO DE COMPONENTES PLACA DEFINITIVA

Cantidad	Componente	Valor / Fabricante	Dimensiones [mm] (ancho x largo x alto)	Montaje	Precio [€]
1	Acelerómetro digital ADXL345	Analog Devices	3 x 5 x 1	superficial	5,62
1	Microcontrolador PIC16LF876A	Microchip	5,3 x 10,2 x 1,75	superficial	5,4
1	Módulo Bluetooth RN42	Roving Networks	13,4 x 25,8 x 2	superficial	12,1
1	Regulador de tensión TPS62203	Texas Instrument	1,6 x 2,9 x 1,15	superficial	2,26
1	Oscilador Cristal SG310SCF	4 Mhz / Epson	2,5 x 3,2 x 1	superficial	3,05
1	Diodo LED rojo HSMC-A431-X90M1	Avago Technologies	3,2 x 2,8 x 3,87	superficial	0,31
1	Diodo LED verde HSMM-A430-X90M2	Avago Technologies	3,2 x 2,8 x 3,87	superficial	0,96
1	Condensador (huella 0603)	10 uF	0,8 x 1,6 x 0,8	superficial	0,18
1	Condensador (huella 0402)	4,7 uF	0,5 x 1 x 0,5	superficial	0,06
5	Condensador (huella 0402)	100 nF	0,5 x 1 x 0,5	superficial	0,01 / ud
1	Resistencia (huella 0402)	10 k	0,5 x 1 x 0,35	superficial	0,03
1	Resistencia (huella 0402)	1 k	0,5 x 1 x 0,35	superficial	0,03
2	Resistencia (huella 0402)	220	0,5 x 1 x 0,35	superficial	0,03 / ud
1	Resistencia (huella 0402)	10	0,5 x 1 x 0,35	superficial	0,14
1	Inductancia (huella 0603)	10 uH	0,9 x 1,7 x 0,9	superficial	0,11
1	Conector microUSB hembra	TE Connectivity	8,8 x 5,6 x 3	superficial	0,99
1	Interruptor deslizante MMP121-R	Knitter Switch	9,6 x 3,5(+2) x 3,5(+3)	pasante	0,85
3	Portapilas botón para PCB	2996 / Keystone	13,2 x 12,7 x 6,1	superficial	0,86 / ud
3	Batería tipo botón SR44	RS	Diam=11,6 – Alt=5,4	-	1,66 / ud

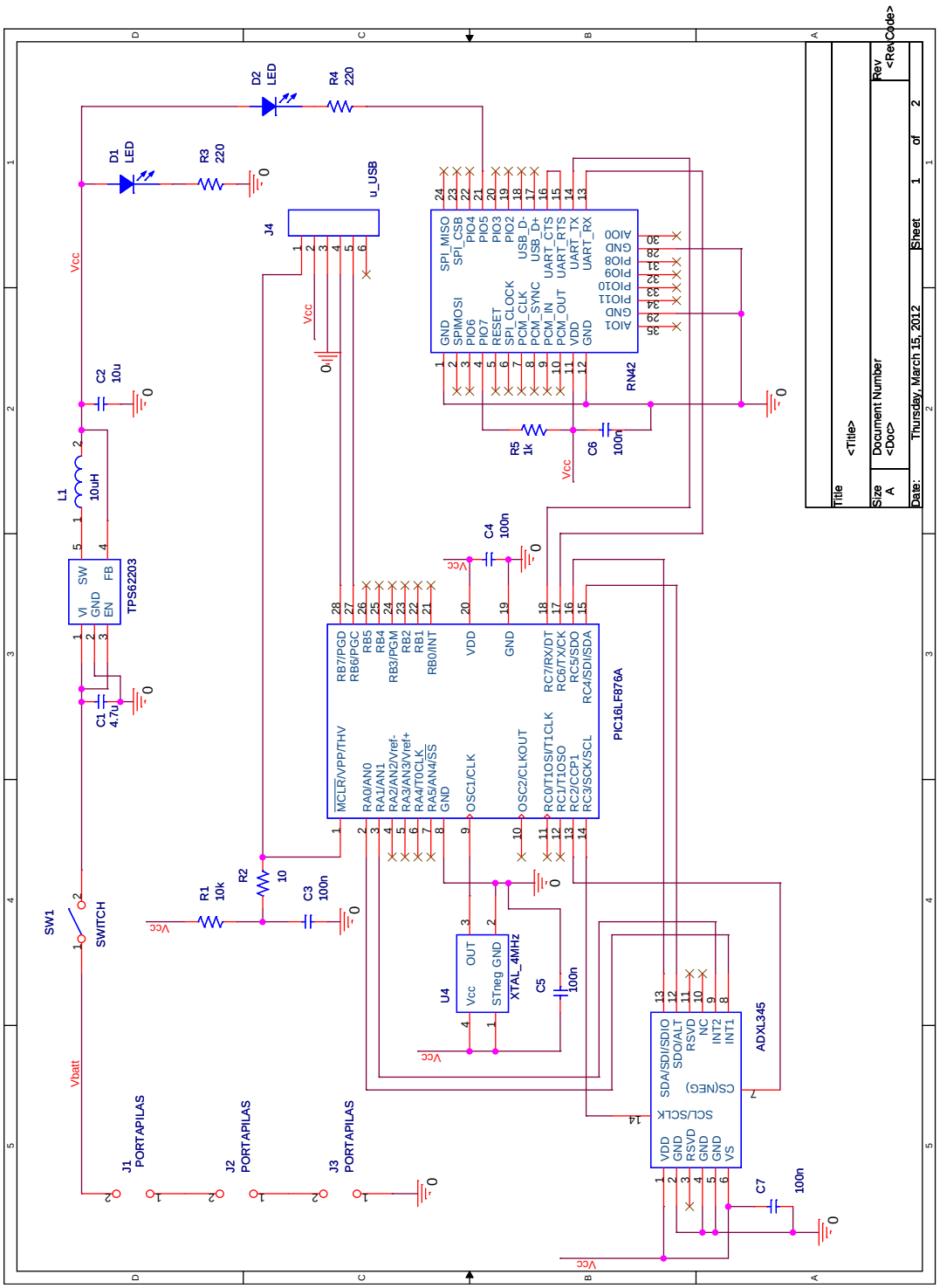
Apéndice C

Esquemáticos

C.1. Placa experimental



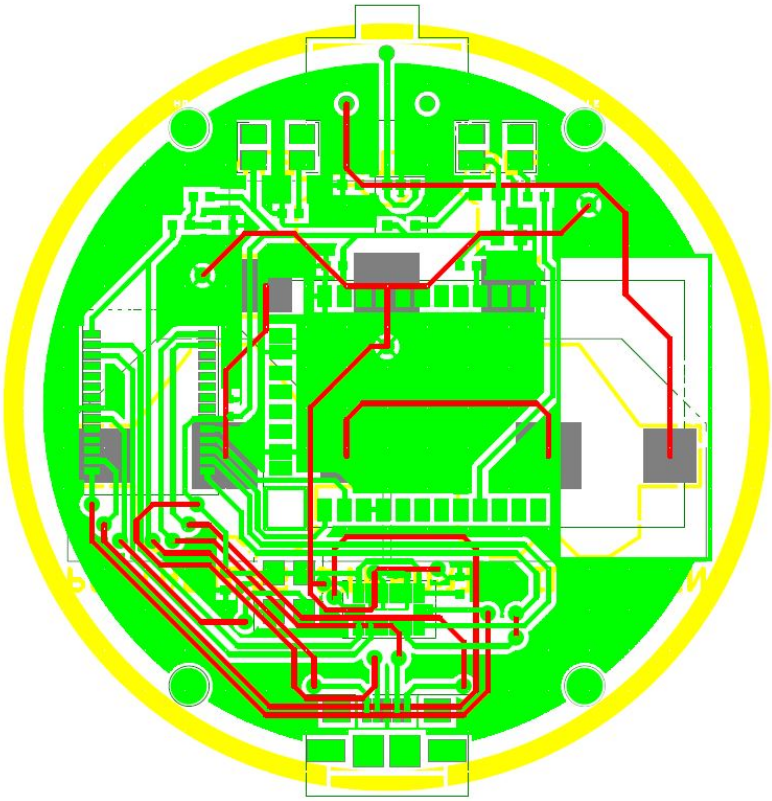
C.2. Placa definitiva



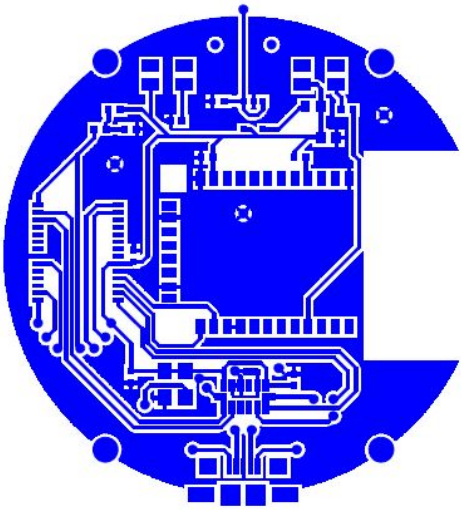
Apéndice D

Circuito impreso. Fotolitos

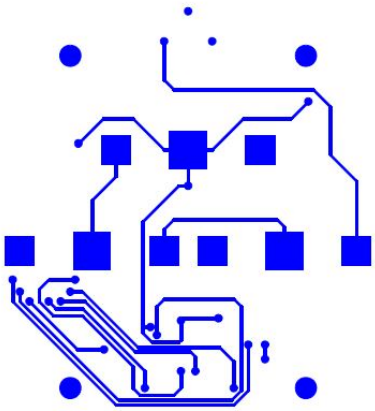
D.1. Layout Completo



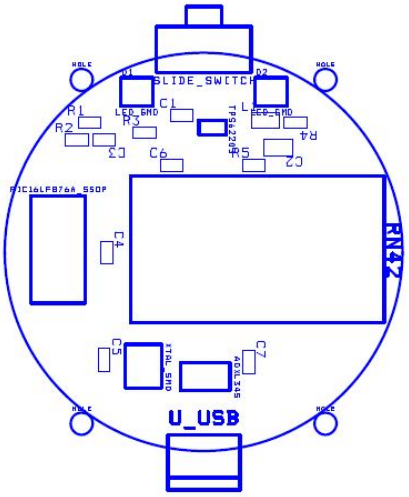
D.2. Capa Top



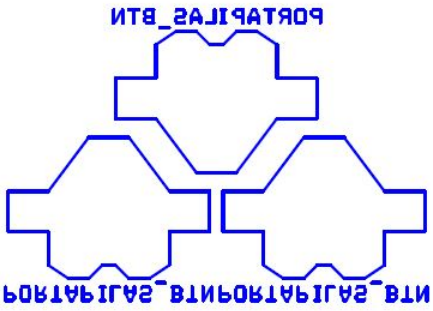
D.3. Capa Bottom



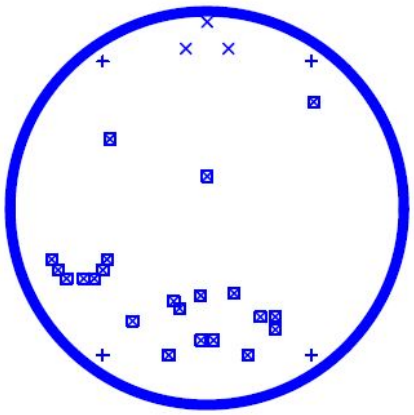
D.4. Capa SSTOP



D.5. Capa SSBOT



D.6. Drill Chart



DRILL CHART				
SYM	DIAM	TOL	QTY	NOTE
×	0.024		3	
⊠	0.028		22	
+	0.090		4	
TOTAL			29	

Apéndice E

Hojas de Características

E.1. ADXL345

3-Axis, ±2 *g*/±4 *g*/±8 *g*/±16 *g* Digital Accelerometer

ADXL345

FEATURES

Ultralow power: as low as 23 µA in measurement mode and 0.1 µA in standby mode at V_S = 2.5 V (typical)
Power consumption scales automatically with bandwidth
User-selectable resolution
Fixed 10-bit resolution
Full resolution, where resolution increases with *g* range, up to 13-bit resolution at ±16 *g* (maintaining 4 m*g*/LSB scale factor in all *g* ranges)
Patent pending, embedded memory management system with FIFO technology minimizes host processor load
Single tap/double tap detection
Activity/inactivity monitoring
Free-fall detection
Supply voltage range: 2.0 V to 3.6 V
I/O voltage range: 1.7 V to V_S
SPI (3- and 4-wire) and I²C digital interfaces
Flexible interrupt modes mappable to either interrupt pin
Measurement ranges selectable via serial command
Bandwidth selectable via serial command
Wide temperature range (−40°C to +85°C)
10,000 *g* shock survival
Pb free/RoHS compliant
Small and thin: 3 mm × 5 mm × 1 mm LGA package

APPLICATIONS

Handsets
Medical instrumentation
Gaming and pointing devices
Industrial instrumentation
Personal navigation devices
Hard disk drive (HDD) protection

GENERAL DESCRIPTION

The ADXL345 is a small, thin, ultralow power, 3-axis accelerometer with high resolution (13-bit) measurement at up to ±16 *g*. Digital output data is formatted as 16-bit twos complement and is accessible through either a SPI (3- or 4-wire) or I²C digital interface.

The ADXL345 is well suited for mobile device applications. It measures the static acceleration of gravity in tilt-sensing applications, as well as dynamic acceleration resulting from motion or shock. Its high resolution (3.9 m*g*/LSB) enables measurement of inclination changes less than 1.0°.

Several special sensing functions are provided. Activity and inactivity sensing detect the presence or lack of motion by comparing the acceleration on any axis with user-set thresholds. Tap sensing detects single and double taps in any direction. Free-fall sensing detects if the device is falling. These functions can be mapped individually to either of two interrupt output pins. An integrated, patent pending memory management system with a 32-level first in, first out (FIFO) buffer can be used to store data to minimize host processor activity and lower overall system power consumption.

Low power modes enable intelligent motion-based power management with threshold sensing and active acceleration measurement at extremely low power dissipation.

The ADXL345 is supplied in a small, thin, 3 mm × 5 mm × 1 mm, 14-lead, plastic package.

FUNCTIONAL BLOCK DIAGRAM

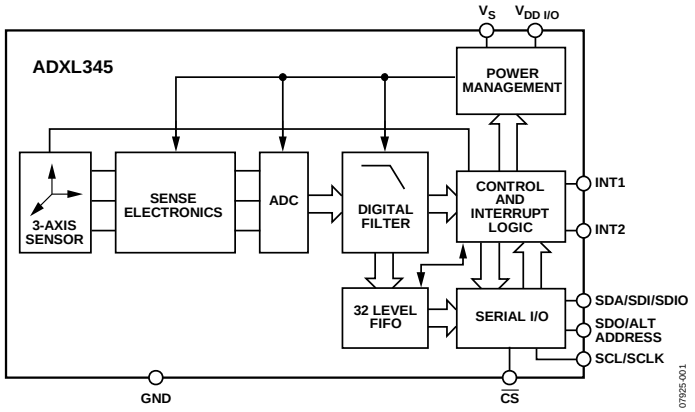


Figure 1.

One Technology Way, P.O. Box 9106, Norwood, MA 02062-9106, U.S.A.
Tel: 781.329.4700 **www.analog.com**
Fax: 781.461.3113

E.2. ADXL345 - Placa de Evaluación

3-Axis, ±2 *g*/±4 *g*/±8 *g*/±16 *g* Digital Accelerometer Evaluation Board

EVAL-ADXL345Z

GENERAL DESCRIPTION

The EVAL-ADXL345Z is a simple evaluation board that allows quick evaluation of the performance of the ADXL345 3-axis digital accelerometer. The EVAL-ADXL345Z has two sets of 0.1 inch spaced vias, for population of a 4-pin and 5-pin header, for access to all power and signal lines. The vias or headers allow the evaluation board to be attached to a prototyping board (breadboard) or attached to the PCB in an existing system. Four holes are provided for mechanical attachment of the EVAL-ADXL345Z to the application. An external host processor is required for communication to the part.

The dimensions of the EVAL-ADXL345Z are 20 mm × 20 mm with mounting holes set 15 mm × 15 mm at the corners of the printed circuit board (PCB).

CIRCUIT DESCRIPTION

The schematic of the EVAL-ADXL345Z is shown in Figure 1. See the ADXL345 data sheet for configuration of the accelerometer after it is connected to the application host processor.

The PCB layout of the EVAL-ADXL345Z is shown in Figure 2. The EVAL-ADXL345Z has three factory installed capacitors for bypass: two 100 nF capacitors and a 10 µF capacitor. C1 and C2 are V_S bypass capacitors to reduce analog supply noise and C3, located between $V_{DD\ I/O}$ and GND, is provided to reduce digital clocking noise.

HANDLING CONSIDERATIONS

The EVAL-ADXL345Z is not reverse polarity protected. Reversing the V_S or $V_{DD\ I/O}$ supply and GND pins can cause damage to the ADXL345.

Dropping the EVAL-ADXL345Z on a hard surface can generate several thousand *g*'s of acceleration, which may exceed the data sheet absolute maximum limits. See the ADXL345 data sheet for more information.

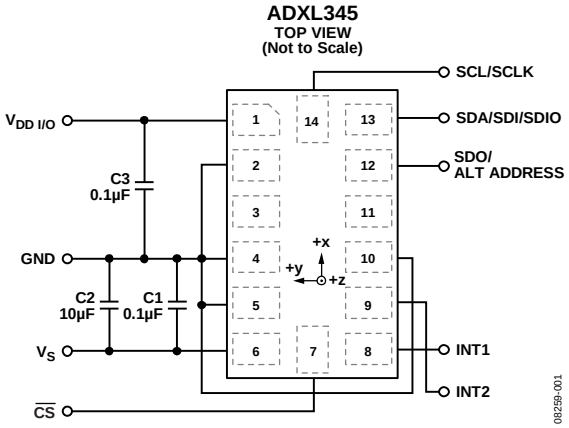


Figure 1. Schematic

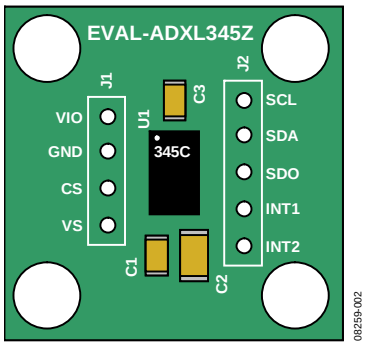


Figure 2. Physical Layout

Rev. 0

One Technology Way, P.O. Box 9106, Norwood, MA 02062-9106, U.S.A.
Tel: 781.329.4700 **www.analog.com**
Fax: 781.461.3113

E.3. PIC16LF876A

PIC16F87XA

28/40-Pin Enhanced FLASH Microcontrollers

Devices Included in this Data Sheet:

- PIC16F873A
- PIC16F874A
- PIC16F876A
- PIC16F877A

High Performance RISC CPU:

- Only 35 single word instructions to learn
- All single cycle instructions except for program branches, which are two-cycle
- Operating speed: DC - 20 MHz clock input
  DC - 200 ns instruction cycle
- Up to 8K x 14 words of FLASH Program Memory, Up to 368 x 8 bytes of Data Memory (RAM), Up to 256 x 8 bytes of EEPROM Data Memory
- Pinout compatible to other 28-pin or 40/44-pin PIC16CXXX and PIC16FXXX microcontrollers

Peripheral Features:

- Timer0: 8-bit timer/counter with 8-bit prescaler
- Timer1: 16-bit timer/counter with prescaler, can be incremented during SLEEP via external crystal/clock
- Timer2: 8-bit timer/counter with 8-bit period register, prescaler and postscaler
- Two Capture, Compare, PWM modules
  - Capture is 16-bit, max. resolution is 12.5 ns
  - Compare is 16-bit, max. resolution is 200 ns
  - PWM max. resolution is 10-bit
- Synchronous Serial Port (SSP) with SPI™ (Master mode) and I^2C™ (Master/Slave)
- Universal Synchronous Asynchronous Receiver Transmitter (USART/SCI) with 9-bit address detection
- Parallel Slave Port (PSP) 8-bits wide, with external $\overline{RD}$, $\overline{WR}$ and $\overline{CS}$ controls (40/44-pin only)
- Brown-out detection circuitry for Brown-out Reset (BOR)

Analog Features:

- 10-bit, up to 8 channel Analog-to-Digital Converter (A/D)
- Brown-out Reset (BOR)
- Analog Comparator module with:
  - Two analog comparators
  - Programmable on-chip voltage reference (V_{REF}) module
  - Programmable input multiplexing from device inputs and internal voltage reference
  - Comparator outputs are externally accessible

Special Microcontroller Features:

- 100,000 erase/write cycle Enhanced FLASH program memory typical
- 1,000,000 erase/write cycle Data EEPROM memory typical
- Data EEPROM Retention > 40 years
- Self-reprogrammable under software control
- In-Circuit Serial Programming™ (ICSP™) via two pins
- Single supply 5V In-Circuit Serial Programming
- Watchdog Timer (WDT) with its own on-chip RC oscillator for reliable operation
- Programmable code protection
- Power saving SLEEP mode
- Selectable oscillator options
- In-Circuit Debug (ICD) via two pins

CMOS Technology:

- Low power, high speed FLASH/EEPROM technology
- Fully static design
- Wide operating voltage range (2.0V to 5.5V)
- Commercial and Industrial temperature ranges
- Low power consumption

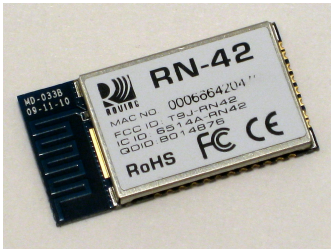
Device	Program Memory		Data SRAM (Bytes)	EEPROM (Bytes)	I/O	10-bit A/D (ch)	CCP (PWM)	MSSP		USART	Timers 8/16-bit	Comparators
	Bytes	# Single Word Instructions						SPI	Master I^2C			
PIC16F873A	7.2K	4096	192	128	22	5	2	Yes	Yes	Yes	2/1	2
PIC16F874A	7.2K	4096	192	128	33	8	2	Yes	Yes	Yes	2/1	2
PIC16F876A	14.3K	8192	368	256	22	5	2	Yes	Yes	Yes	2/1	2
PIC16F877A	14.3K	8192	368	256	33	8	2	Yes	Yes	Yes	2/1	2

E.4. RN42

RN-42/RN-42-N Data Sheet

www.rovingnetworks.com DS-RN42-V1.0 9/22/2010

Class 2 Bluetooth® Module



Features

- Fully qualified Bluetooth 2.1/2.0/1.2/1.1 module
- Bluetooth v2.0+EDR support
- Available with on board chip antenna (RN-42) and without antenna (RN-42-N)
- Postage stamp sized form factor, 13.4mm x 25.8 mm x 2mm (RN-42) and 13.4mm x 20 mm x 2 mm (RN-42-N)
- Low power *(26uA sleep, 3mA connected, 30mA transmit)*
- UART (SPP or HCI) and USB (HCI only) data connection interfaces.
- Sustained SPP data rates - 240Kbps (slave), 300Kbps (master)
- HCI data rates - 1.5Mbps sustained, 3.0Mbps burst in HCI mode
- Embedded Bluetooth stack profiles included *(requires no host stack)*: GAP, SDP, RFCOMM and L2CAP protocols, with SPP and DUN profile support.
- Bluetooth SIG certified
- Castellated SMT pads for easy and reliable PCB mounting
- Certifications: FCC, ICS, CE
- Environmentally friendly, RoHS compliant

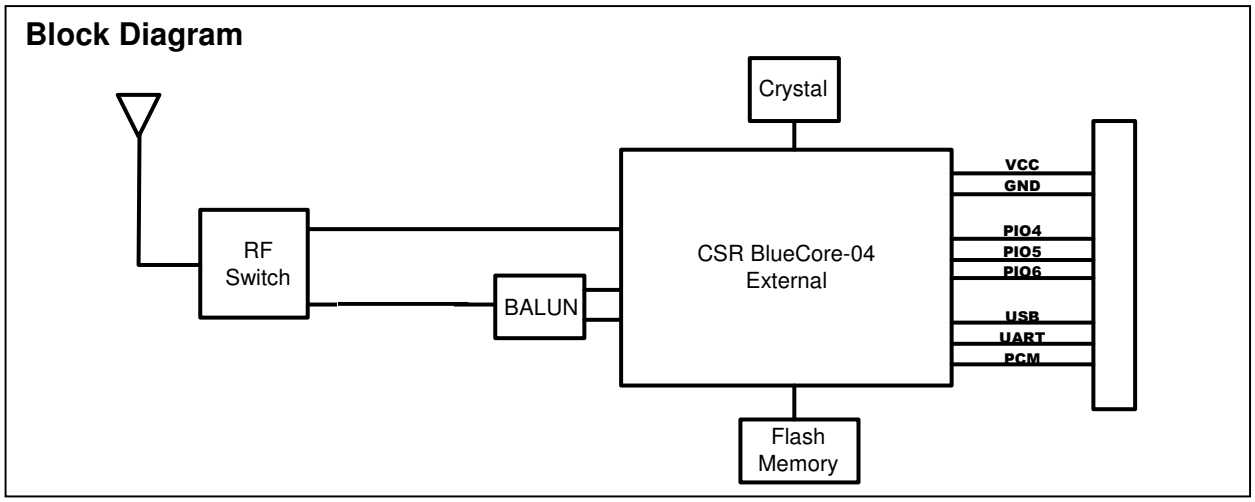
Applications

- Cable replacement
- Barcode scanners
- Measurement and monitoring systems
- Industrial sensors and controls
- Medical devices
- Barcode readers
- Computer accessories

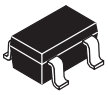
Description

The RN42 is a small form factor, low power, highly economic Bluetooth radio for OEM's adding wireless capability to their products. The RN42 supports multiple interface protocols, is simple to design in and fully certified, making it a complete embedded Bluetooth solution. The RN 42 is functionally compatible with RN 41. With its high performance on chip antenna and support for Bluetooth® Enhanced Data Rate (EDR), the RN42 delivers up to 3 Mbps data rate for distances to 20M. The RN-42 also comes in a package with no antenna (RN-42-N). Useful when the application requires an external antenna, the RN-42-N is shorter in length and has RF pads to route the antenna signal.

Block Diagram



E.5. TPS62203



HIGH-EFFICIENCY, SOT23 STEP-DOWN, DC-DC CONVERTER

FEATURES

- **High Efficiency Synchronous Step-Down Converter With up to 95% Efficiency**
- **2.5-V to 6-V Input Voltage Range**
- **Adjustable Output Voltage Range From 0.7 V to V_I**
- **Fixed Output Voltage Options Available**
- **Up to 300 mA Output Current**
- **1-MHz Fixed Frequency PWM Operation**
- **Highest Efficiency Over Wide Load Current Range Due to Power Save Mode**
- **15-µA Typical Quiescent Current**
- **Soft Start**
- **100% Duty Cycle Low-Dropout Operation**
- **Dynamic Output-Voltage Positioning**
- **Available in a 5-Pin SOT23 Package**

APPLICATIONS

- **PDAs and Pocket PC**
- **Cellular Phones, Smart Phones**
- **Low Power DSP Supply**
- **Digital Cameras**
- **Portable Media Players**
- **Portable Equipment**

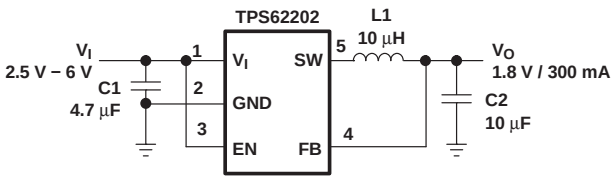
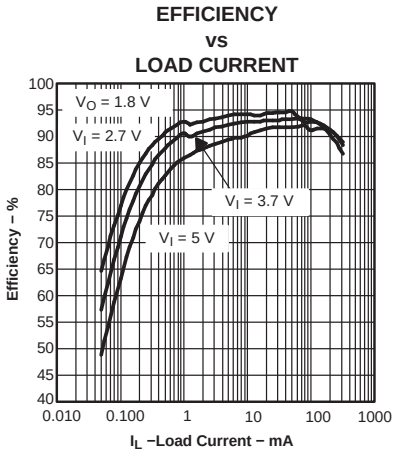


Figure 1. Typical Application (Fixed Output Voltage Version)

DESCRIPTION

The TPS6220x devices are a family of high-efficiency synchronous step-down converters ideally suited for portable systems powered by 1-cell Li-Ion or 3-cell NiMH/NiCd batteries. The devices are also suitable to operate from a standard 3.3-V or 5-V voltage rail.

With an output voltage range of 6 V down to 0.7 V and up to 300 mA output current, the devices are ideal to power low voltage DSPs and processors used in PDAs, pocket PCs, and smart phones. Under nominal load current, the devices operate with a fixed switching frequency of typically 1 MHz. At light load currents, the part enters the power save mode operation; the switching frequency is reduced and the quiescent current is typically only 15 µA; therefore, it achieves the highest efficiency over the entire load current range. The TPS6220x needs only three small external components. Together with the SOT23 package, a minimum system solution size is achieved. An advanced fast response voltage mode control scheme achieves superior line and load regulation with small ceramic input and output capacitors.



E.6. Cristal 4MHz SMD

CRYSTAL OSCILLATOR

SPXO

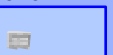
SG - 310 series

- Frequency range : 2 MHz to 80 MHz
- Supply voltage : 1.8 V Typ. / 2.5 V Typ. / 3.3 V Typ.
- Current consumption : 1.5 mA Typ. (SEF1.8 V No load condition 48 MHz)
- Function : Standby($\overline{ST}$)
- Thickness : 1.05 mm Typ.

Pb Free RoHS Compliant

Product Number (please contact us)
Q33310xx0xxxx00

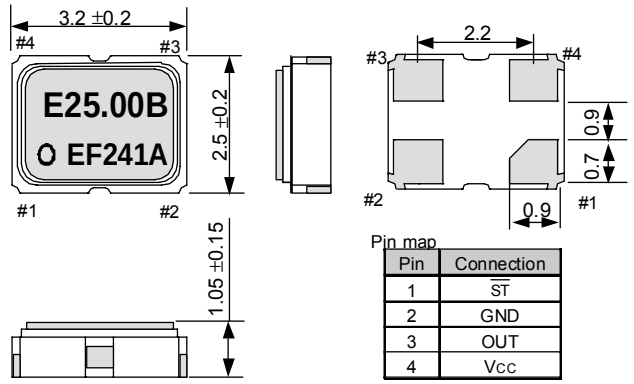
Actual size



Specifications (characteristics)

Item		Symbol	Specifications					Remarks	
			SG-310 SEF	SG-310 SDF	SG-310 SCF	SG-310 SDN	SG-310 SCN		
Output frequency range		f0	2.000 MHz to 48.000 MHz			3.000 MHz to 80.000 MHz			
Supply voltage		Vcc	1.8 V Typ. 1.6 V to 2.2 V	2.5 V Typ. 2.2 V to 3.0 V	3.3 V Typ. 2.7 V to 3.6 V	2.5 V Typ. 2.2 V to 2.7 V	3.3 V Typ. 2.7 V to 3.6 V		
Temperature range	Storage temperature	T_stg	-40 °C to +125 °C					Store as bare product after unpacking	
	Operating temperature	T_use	-40 °C to +85 °C						
Frequency tolerance		f_tol	B: ±50 × 10^{-6}, C: ±100 × 10^{-6}			B: ±50 × 10^{-6}, C: ±100 × 10^{-6}		-20 °C to +70 °C	
			M: ±100 × 10^{-6}			M: ±100 × 10^{-6}		-40 °C to +85 °C	
			–			D:±20 × 10^{-6} ,S:±25 × 10^{-6}		-20 °C to +70 °C	Vcc ±10 %
			–			R:±25 × 10^{-6}		-30 °C to +85 °C	(3 MHz<f0≤62.5 MHz)
			–			P:±20 × 10^{-6}		-30 °C to +85 °C	Vcc ±5 %
			–			J:±25 × 10^{-6}		-40 °C to +85 °C	(3 MHz<f0≤62.5 MHz)
Current consumption		Icc	1.5 mA Max.	1.5 mA Max.	1.5 mA Max.	4.0 mA Max.	5.0 mA Max.	No load condition, 2 MHz≤f0≤ 4 MHz	
			1.5 mA Max.	1.5 mA Max.	2.0 mA Max.			No load condition, 4 MHz<f0≤ 8 MHz	
			1.5 mA Max.	2.0 mA Max.	2.5 mA Max.			No load condition, 8 MHz<f0≤16 MHz	
			2.0 mA Max.	2.0 mA Max.	2.5 mA Max.			No load condition, 16 MHz<f0≤25 MHz	
			2.0 mA Max.	2.5 mA Max.	3.5 mA Max.			No load condition, 25 MHz<f0≤33 MHz	
			3.0 mA Max.	3.5 mA Max.	4.5 mA Max.			No load condition, 33 MHz<f0≤48 MHz	
			–			6.0 mA Max.	7.0 mA Max.	No load condition, 48 MHz<f0≤80 MHz	
Stand-by current		I_std	0.7 µA Max. (0.2 µA Typ.)	1.5 µA Max. (0.5 µA Typ.)	2.0 µA Max. (1.0 µA Typ.)	10 µA Max.		$\overline{ST}$ =GND	
Symmetry		SYM	45 % to 55 %	45 % to 55 %	45 % to 55 %	45 % to 55 %		2 MHz≤f0≤16 MHz	50 % Vcc level L_CMOS ≤ 15 pF
			40 % to 60 %	40 % to 60 %				16 MHz<f0≤40 MHz	
								40 MHz<f0≤80 MHz	
High output voltage		VOH	90 % Vcc Min.					IOH=-3 mA	
Low output voltage		VOL	10 % Vcc Max.					IOL= 3 mA	
Output load condition (CMOS)		L_CMOS	15 pF Max.						
Output enable / disable input voltage		VIH	80 % Vcc Min.			70 % Vcc Min.		$\overline{ST}$ terminal	
		VIL	20 % Vcc Max.			30 % Vcc Max.			
Rise time / Fall time		tr/ tf	4 ns Max.					20 % Vcc to 80 % Vcc level, L_CMOS=15 pF	
Start-up time		t_str	10 ms Max.			2 ms Max.		t=0 at 90 % Vcc	
Frequency aging		f_aging	±5 × 10^{-6} / year Max.			±3 × 10^{-6} / year Max.		+25 °C, First year,Vcc=1.8 V, 2.5 V, 3.3 V	
			–			±10 × 10^{-6} Max.		+25 °C, 10 years	

External dimensions (Unit:mm)

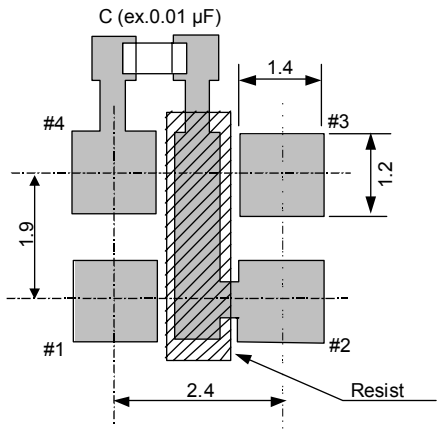


Pin map

Pin	Connection
1	$\overline{ST}$
2	GND
3	OUT
4	Vcc

Note.
$\overline{ST}$ pin = HIGH or "open" : Specified frequency output.
$\overline{ST}$ pin = LOW : Output is high impedance, oscillation stops.

Footprint (Recommended) (Unit:mm)



E.7. LED SMD

HSMx-A43x-xxxxx
Surface Mount LED Indicator

Data Sheet

Lead (Pb) Free
RoHS 6 fully
compliant

Description

The Power PLCC-4 SMT LED with Lens are high-performance PLCC-4 package size SMT LEDs targeted mainly in Automotive & Electronics Signs and Signals (ESS) markets. These top-mount single-chip packages with focused radiation offer high brightness in beam direction and are excellent for interior automotive, indoor and outdoor sign and industrial applications. With additional lens in 30° variants, these products are especially fitting to applications for traffic lights, CHMSL and displays.

The PLCC-4 package family is able to dissipate heat better compared to the PLCC-2 packages. In proportion to this increase in driving current, this family of LEDs is able to produce higher light output compared to the conventional PLCC-2 SMT LEDs.

As an extension of the standard flat top PLCC-4 SMT LEDs, the Power PLCC-4 with Lens device is able to provide focused beams within narrow viewing angles (30°) meeting the market's requirements for focused radiation and high brightness in beam directions.

The Power PLCC-4 SMT LED with 30° is ideal for panel, push button, or general backlighting in automotive interior & exterior, sign, office equipment, industrial equipment and home appliances applications. This package design coupled with careful selection of component materials allow the Power PLCC-4 SMT LED with Lens to perform with higher reliability in a larger temperature range -40°C to 100°C. This high reliability feature is crucial to allow the Power PLCC-4 SMT LED with Lens to do well in harsh environments such as its target Automotive & ESS markets. The Power PLCC4 SMT LED with Lens package is also designed to be compatible with both IR-solder re-flow and through-the-wave soldering.

Features

- Industry Standard PLCC-4
- High reliability LED package
- High brightness using AlInGaP and InGaN dice technologies
- High optical efficiency
- Narrow Viewing angle at 30°
- Available in 8mm carrier tape on 7-inch reel
- Compatible with both IR and TTW soldering process

Applications

Interior automotive

- Instrument panel backlighting
- Central console backlighting
- Cabin backlighting
- Navigation and audio system
- Dome lighting
- Push button backlighting

Exterior automotive

- Turn signals
- CHMSL
- Rear Combination Lamp
- Side repeaters

Electronic signs and signals

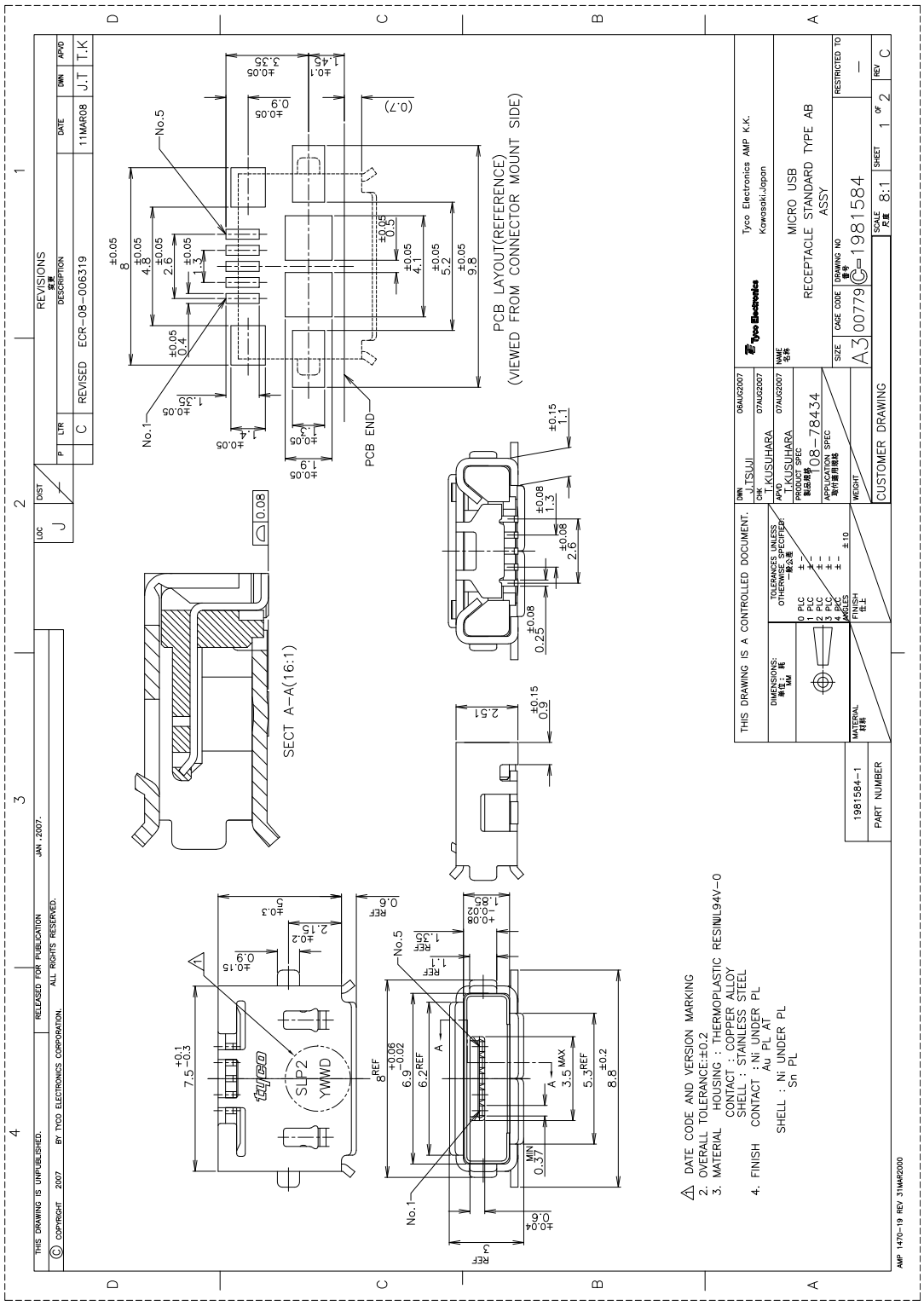
- Interior and exterior full color sign
- Variable message sign
- Garden lighting

Office automation, home appliances, industrial equipment

- Front panel backlighting
- Push button backlighting
- Display backlighting

CAUTION: *HSMN, HSMM-A43x-xxxxx LEDs are Class 2 ESD sensitive. Please observe appropriate precautions during handling and processing. Refer to Avago Application Note AN-1142 for additional details.*

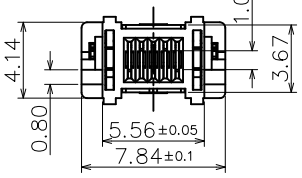
E.8. Conector microUSB hembra

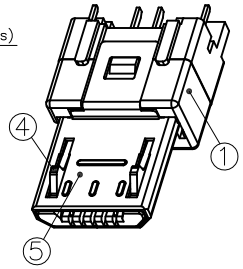

PCB LAYOUT(REFERENCE)
(VIEWED FROM CONNECTOR MOUNT SIDE)
PCB END
No.1
No.5
SECT A–A(16:1)
SLP2
YWWD
DATE CODE AND VERSION MARKING
OVERALL TOLERANCE:±0.2
MATERIAL HOUSING : THERMOPLASTIC RESIN UL94V-0
CONTACT : COPPER ALLOY
SHELL : STAINLESS STEEL
FINISH CONTACT : Ni UNDER PL
Au PL AT
SHELL : Ni UNDER PL
Sn PL
REVISIONS
REVISED ECR-08-006319
11MAR08
Tyco Electronics AMP K.K.
Kawasaki.Japan
MICRO USB
RECEPTACLE STANDARD TYPE AB
ASSY
1981584
CUSTOMER DRAWING
THIS DRAWING IS A CONTROLLED DOCUMENT.
THIS DRAWING IS UNPUBLISHED. RELEASED FOR PUBLICATION JAN-2007.
COPYRIGHT 2007 BY TYCO ELECTRONICS CORPORATION. ALL RIGHTS RESERVED.
AMP 1470-19 REV 31MAR2000

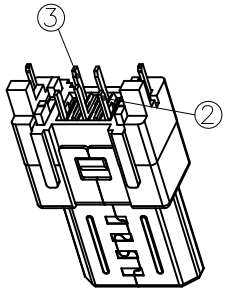
E.9. Conector microUSB macho

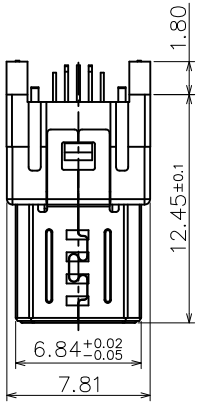
Este componente sólo ha sido utilizado en el adaptador RJ11–microUSB.

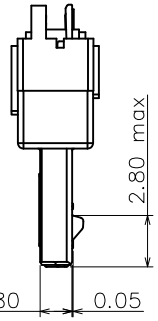
Part Number
10104109-0001LF

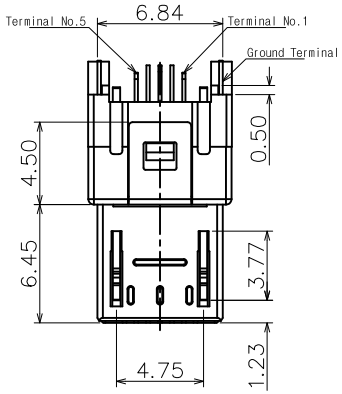


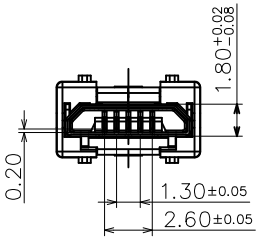


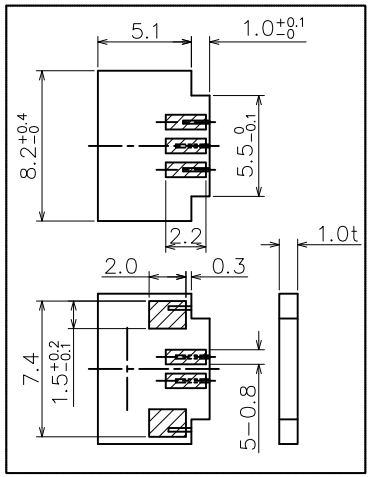






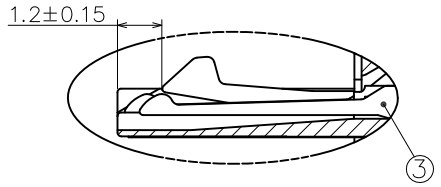




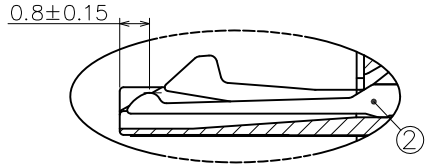


*PCB ref.
(Tolerance ±0.05)

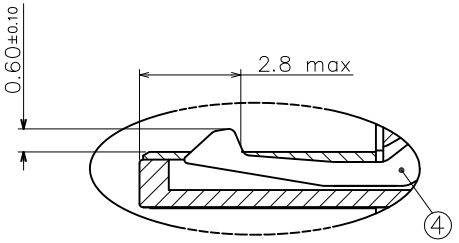
-Note-
1. Material & finish: See table-1
2. Packing in tray



Plug Contact -A Point -2EA (No.2, No.4)
Plug Contact -B Point -1EA (No.3)



Plug Contact -B Point -2EA (No.1, No.5)



Plug Contact -LOCK TML' Point -2EA

(Table-1)

No.	Part Name	Material	Finish	Remarks
1	Plug Housing	Thermoplastic	black	
2	Plug Contact -B	Copper alloy	Selective Gold Plated	
3	Plug Contact -A	Copper alloy	Selective Gold Plated	
4	Lock Plate	Sustainess steel		
5	Plug Shield	Sustainess steel	Ni Plated	

mat'l. code			
ltr	ecn no	dr	date
A	T10-0002	J.H	01/07/10

tolerances unless otherwise specified	
linear	.X ± 0.38
	.XX ± 0.25
	.XXX ± 0.10
angles	0° ±2°

dr	JASON HSU	01/07/10
engr	JASON HSU	01/07/10
chr	GARY HSIEH	01/07/10
appd	JOSEPH HSIA	01/07/10

sheet index	revision	
sheet	A	
	1	

CUSTOMER COPY

projection

MM

scale 1:1

FCI www.fciconnect.com

title: MICRO-USB B-TYPE PLUG

product family: USB

code: TWN

size: A4 | dwg no: 10104109

sheet: 1 of 1

cage code

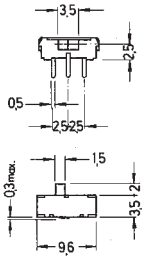
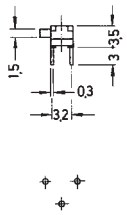
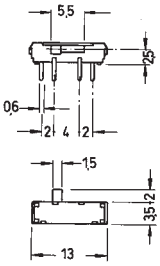
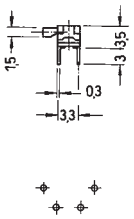
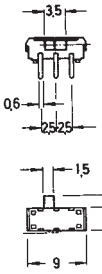
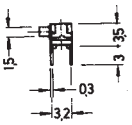
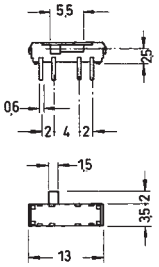
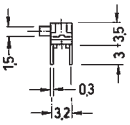
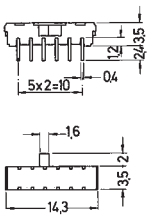
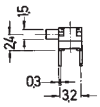
ACAD

PDM: Rev:A

STATUS: Released

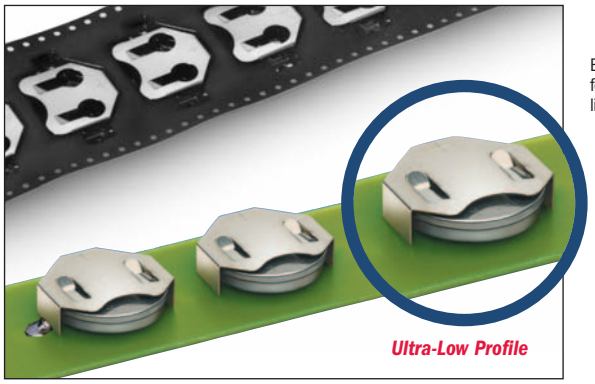
Printed: Jun 02, 2010

E.10. Interruptor deslizante

Models:	Switching functions					
MMP 121 – R	on	on				
MMP 131 – R	on	on	on			
MMP 221 – R	on	on				
MMP 231 – R	on	on	on			
MMP 421 – R	on	on				

E.11. Portapilas botón para PCB

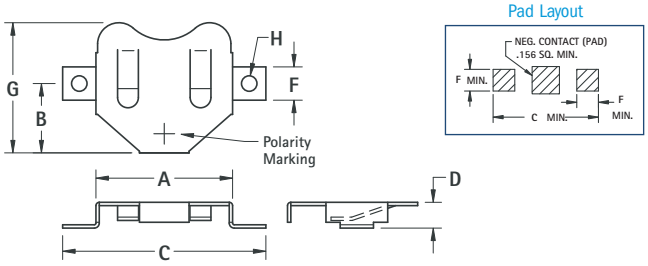
SURFACE & THRU HOLE MOUNT COMPACT CONTACTS



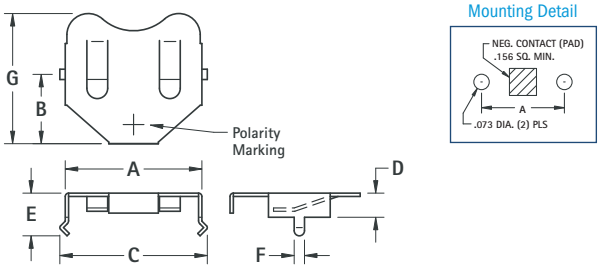
AVAILABLE ON TAPE AND REEL

Economical and reliable retainer contacts are designed for quick and easy battery installation and replacement. Perfect for memory back-up applications. Surface and Thru Hole mount retainer contacts are available for 6.8mm to 24mm lithium coin cell batteries.

- Low profile for densely packed PCBs
- Retains cell securely, withstands shock & vibrations
- Reliable dual spring contacts assure low contact resistance
- Polarity clearly marked (+)
- Stable thru mount legs for increased strength
- Balanced & lightweight for reliable pick-up & placement
- Compatible with vacuum and mechanical pick & place assembly systems
- "Flow-Hole" SMT solder tail design for increased joint strength
- Viewable solder tails assures reliable joint inspection
- All conductive polystyrene carrier tape meets ANSI/EIA- 481 standard



DWG. 1 - SURFACE MOUNT (SMT) **MATERIAL:** .010 (.25) thick Phosphor Bronze, Tin Nickel Plate



DWG. 2 - THRU HOLE MOUNT (THM) **MATERIAL:** .010 (.25) thick Phosphor Bronze, Tin Nickel Plate

CAT. NO.	ON TAPE & REEL	DWG. NO.	MTG. TYPE	A	B	C	D	E	F	G	H DIA.	CELL DIA.	BATTERY REFERENCE	TAPE & REEL SPEC'S
2996	2996TR	1	SMT	.520 (13.2)	.265 (6.7)	.700 (17.8)	.235 (6.0)	–	.125 (3.2)	.475 (12.1)	.053 (1.3)	11.6mm	SR44, LR44, 357, MS76, PX76A, G-13, 303, V313	32mm wide; 16mm pitch; 13 inch reel (600 pieces per reel)
2997	—	2	THM	.520 (13.2)	.265 (6.7)	.585 (14.9)	.235 (6.0)	.323 (8.2)	.062 (1.6)	.475 (12.1)	–			
3000	3000TR	1	SMT	.520 (13.2)	.265 (6.7)	.745 (18.9)	.125 (3.2)	–	.125 (3.2)	.475 (12.1)	.047 (1.2)	12mm	BR1216, CR1216, BR1220, CL1220, CR1220, BR1225	32mm wide; 16mm pitch; 13 inch reel (1000 pieces per reel)
3001	—	2	THM	.520 (13.2)	.265 (6.7)	.585 (14.9)	.118 (3.0)	.210 (5.3)	.062 (1.6)	.475 (12.1)	–			
3012	3012TR	1	SMT	.664 (16.9)	.344 (8.7)	.914 (23.2)	.156 (4.0)	–	.125 (3.2)	.594 (15.1)	.047 (1.2)	16mm	BR1616, CR1612, CR1616, CR1620, CR1632	44mm wide; 24mm pitch; 13 inch reel (600 pieces per reel)
3013	—	2	THM	.664 (16.9)	.344 (8.7)	.723 (18.4)	.156 (4.0)	.265 (6.7)	.062 (1.6)	.594 (15.1)	–			
3002	3002TR	1	SMT	.831 (21.1)	.416 (10.6)	1.210 (30.7)	.156 (4.0)	–	.200 (5.1)	.782 (19.9)	.093 (2.4)	20mm	BR2016, CR2016, DL2016, BR2020, CL2020, BR2025, CR2025, DL2025, DR2032, CR2032, DL2032	44mm wide; 24mm pitch; 13 inch reel (500 pieces per reel)
3003	—	2	THM	.831 (21.1)	.416 (10.6)	.890 (22.6)	.156 (4.0)	.265 (6.7)	.062 (1.6)	.782 (19.9)	–			
3004	3004TR	1	SMT	.939 (23.9)	.475 (12.1)	1.339 (34.0)	.156 (4.0)	–	.200 (5.1)	.832 (21.1)	.093 (2.4)	23mm	BR2320, CL2330, CR2320, BR2325, CR2325, DL2325	56mm wide; 32mm pitch; 13 inch reel (400 pieces per reel)
3005	—	2	THM	.939 (23.9)	.475 (12.1)	1.004 (25.5)	.156 (4.0)	.265 (6.7)	.062 (1.6)	.832 (21.1)	–			
3010	3010TR	1	SMT	.939 (23.9)	.475 (12.1)	1.339 (34.0)	.240 (6.1)	–	.200 (5.1)	.819 (20.8)	.093 (2.4)	23mm	CR2354, DL2354	56mm wide; 32mm pitch; 13 inch reel (200 pieces per reel)
3011	—	2	THM	.939 (23.9)	.475 (12.1)	1.004 (25.5)	.240 (6.1)	.351 (8.9)	.062 (1.6)	.819 (20.8)	–			
3006	3006TR	1	SMT	1.000 (25.4)	.506 (12.9)	1.400 (35.6)	.156 (4.0)	–	.200 (5.1)	.918 (23.3)	.093 (2.4)	24mm	CR2430, DL2430	56mm wide; 32mm pitch; 13 inch reel (400 pieces per reel)
3007	—	2	THM	1.000 (25.4)	.506 (12.9)	1.055 (26.8)	.156 (4.0)	.265 (6.7)	.062 (1.6)	.918 (23.3)	–			
3008	3008TR	1	SMT	1.000 (25.4)	.506 (12.9)	1.400 (35.6)	.230 (5.8)	–	.200 (5.1)	.890 (22.6)	.093 (2.4)	24mm	CR2450, DL2450	56mm wide; 32mm pitch; 13 inch reel (200 pieces per reel)
3009	—	2	THM	1.000 (25.4)	.506 (12.9)	1.059 (26.9)	.230 (5.8)	.350 (8.9)	.062 (1.6)	.890 (22.6)	–			

For 6.8mm BUTTON CELLS

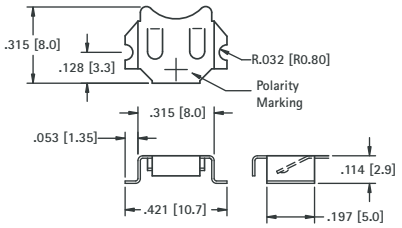
AVAILABLE ON TAPE AND REEL

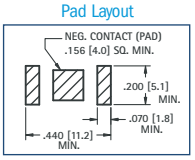
MATERIAL: .010 (.25) thick Phosphor Bronze, Tin Nickel Plate

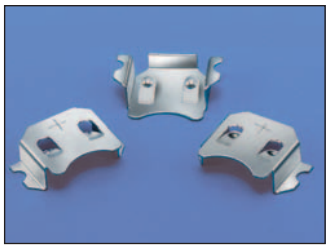
Tape & Reel Spec's: 24mm wide; 12mm pitch; 13 inch reel (1250 pieces per reel)

CAT. NO. 2998 (Bulk)

CAT. NO. 2998TR (Tape and Reel)







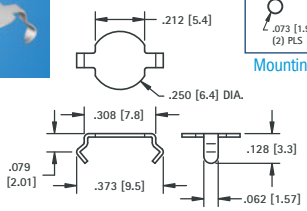
For use with Batteries MC621, V364 and SC621

NEGATIVE BATTERY CONTACTS

• Economical • Space saving • Ideal for self-contained battery compartments • Contact us for modifications and custom design

THRU HOLE MOUNT

• Snap-In

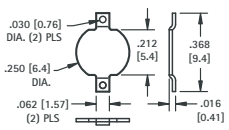


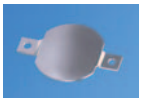
CAT. NO. 2990

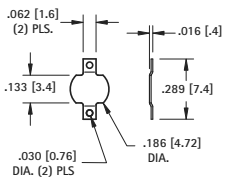
MATERIAL: .008 (.20) thick Phosphor Bronze, Tin Nickel Plate

SURFACE MOUNT

MATERIAL: .008 (.20) thick Brass, Tin Nickel Plate



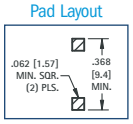




Tape & Reel Spec's: 16mm wide; 8mm pitch; 7 inch reel (1500 pieces per reel)

CAT. NO. 2991 (Bulk)

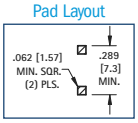
CAT. NO. 2991TR (Tape and Reel)



Tape & Reel Spec's: 16mm wide; 12mm pitch; 7 inch reel (1000 pieces per reel)

CAT. NO. 2992 (Bulk)

CAT. NO. 2992TR (Tape and Reel)



Tel (718) 956-8900 • Fax (718) 956-9040
(800) 221-5510 • kec@keyelco.com
KEYSTONE
RoHS COMPLIANT ~ ISO 9001 CERTIFIED
www.keyelco.com
31-07 20th Road – Astoria, NY 11105-2017

E.12. Batería tipo botón

Ampliar Imagen

La imagen puede no coincidir con la de este producto

Ver productos similares

- Página Principal
- Baterías
- Baterías y Cargadores
- Baterías de botón
- Baterías de Botón, Alcalinas, de Óxido de Plata o Zinc-Aire
- Pulse para ver productos de la misma familia
- Paquete de 5 Pilas SR44 1.55V Alcalinas

Código RS 593-423
Fabricante RS
Página 1943

Disponible en 24 horas

Cantidad	Precio Ud.
5	1,576 €
75	1,496 €
150	1,42 €

Todavía no se ha identificado en RS Online con su nombre de usuario, por lo que puede no estar viendo los precios correspondientes a su empresa. - Entrar

Cantidad 5 **Añadir al pedido**

Comprobar disponibilidad en almacén

Especificaciones

Altura	5,4mm
Anchura	11,6mm
Capacidad	165mAh
Diámetro	11.6mm
Long.	11,6mm
Química	Óxido de plata
Tamaño de batería	Botón
Temperatura de trabajo	-10 → 50°C
Tipo	No recargable

Buscar Si no es exactamente lo que necesita, puede ver más productos seleccionando los atributos adecuados de más arriba y pulsando 'Buscar'